Flux Corrected Transport applied to Hydrodynamics for Heavy Ion Collisions *

Rory Montague Adams

August 08

Thesis presented for the degree of Doctor of Philosophy
In the Department of Physics
University of Cape Town

*Financial assistance of the National Research Foundation(NRF) towards this research is hereby acknowledged. Opinions expressed and conclusions arrived at, are those of the author and are not necessarily to be attriubuted to the NRF.

Abstract

This thesis presents FCTHydro, a ROOT package, and its application to hydrodynamic simulations through the packages RelHydro and Nonideal_xy. These packages aim to provide the broader heavy ion collision community with access to hydrodynamic simulation software which is now accessible from within the primary analysis framework, ROOT. Tests are performed and show how well the high-order, monotone, conservative, positivity preserving routines within FCTHydro simulate hydrodynamic systems with harsh initial conditions. RelHydro illustrates the application of FCTHydro to relativistic systems and Nonideal_xy the application to causal non-ideal hydrodynamic systems. Nonideal_xy is also used to obtain a first order understanding of the effects of the relaxation times in causal non-ideal hydrodynamics.

In addition, a semi-analytic solution for the particle rapidity spectra obtained by combining Landau hydrodynamics and the Cooper-Frye freezeout formalism is given. The results are compared with the Landau Gaussian and a known approximation for midrapidies. The Landau Gaussian provides the best approximation to experimental data.

Furthermore, the chemical freezeout results for preliminary data from AGS for central Au-Au collisions at nominal beam energies 2, 4, 6 and 8 AGeV are shown to agree with the $E/N = 1$ GeV freezeout criteria. These data allow access to a previously unexplored region in the T-μ_B phase space.

Acknowledgements

First and foremost, my thanks go to my wife. Thankyou Tarin, for your continued and unwavering support, patience and encouraging words during the duration of this thesis. It has been wonderful to share my good experiences with you and have you help me through the challenges. To my family, Mom, Dad and Megs, thanks for being there; your support has always been amazing. To my friends, my thanks go to you as well for your support. You no longer have to ask: is it done? I would like to single out Mark Horner, Sarah Blyth and Spencer Wheaton for all their editing, red ink and the helpful discussions - thanks guys.

I would also like to thank my two suppervisors, Prof. Jean Cleymans and Dr. Azwinndini Muronga. Your help and guidance has been very useful. Dr Muronga was instrumental in clearing up certain concepts and ideas in the arena of non-ideal hydrodynamics.

At the beginning of my work I visited Prof. LP Csernai at the University of Bergen and attended his lectures on hydrodynamics and heavy ion collisions. I would like to thank Prof. Csernai and the Department of Physics and Technology at the University of Bergen for their hospitality.

I would also like to thank Dr. Peter Steinberg, a visiting Fullbright Fellow during the initial stages of my PhD work. I was new to the fields of hydrodynamics and heavy ion collisions as my MSc was in field theory. The many colloquia and discussions with Dr. Steinberg were incredibly useful and informative.

While working on my PhD, I also lectured in the Mathematics Department at UCT. I would like thank Alan Rynhoud for all his guidance and friendship during this time. The lecturing helped me get through the darker periods of the research.

Finally I would like to thank the NRF for their financial assistance through the Prestigious Scholarship.

Contents

Chapter 1

Introduction

Mankind has always had a natural curiosity which has many a time led him into trouble, but has also opened avenues to exciting and innovative ideas and discoveries. His curiosity has extended to his environment, matter, and the question, what is it all made of? What are the fundamental particles, the indivisible base constituents? And how do these particles interact? Through the ages he has developed the scientific method of applying his curiosity and through thought and experiment has travelled an interesting road to the current understanding of matter and its basic elements.

The concept of the atom as the building block of matter was first conceptualised in ancient India in the 6^{th} century BC and then again with Democritus' atomos (which means uncuttable) around 450BC. While, our modern atomic theory was started by Boscovich and developed further by the likes of Avogadro, Maxwell, Boltzmann and Dalton. In 1897, Thomson's experiments with cathode rays lead to the discovery of the electron and the realisation that atoms are in fact not fundamental particles but have an underlying structure. Rutherford's examination of the 'gold foil experiment' led to the concept of atoms consisting of a positive nucleus surrounded by a "cloud" of negatively charged electrons. Through further experiments, the nucleus was further understood to consist of protons and neutrons. The concept that the nucleons were fundamental particles of nature was challenged by Feynman and his postulation of the existence of partons [1] in 1969. He realised that protons and neutrons have substructure from the analysis of high energy collisions of protons and electrons at SLAC (Stanford). It was only around the late 1960's and early 1970's, that nucleons were realised to consist of bound states of quarks and gluons [2]. It is our present understanding that quarks and leptons are fundamental particles of nature.

The Standard Model of particle physics is the best current description of the 200^{+} elementary particles and resonances [3] of nature and their interactions (forces). The fundamental particles of the Standard Model consist of 6 quarks, 6 leptons and 5 force-

mediating bosons. The leptons consist of the electron, muon and tauon each with an associated neutrino and the 6 quarks are the up, down, charm, strange, top and bottom quarks. Within the Standard Model, three of the four fundamental forces are mediated by the exchange of the fundamental bosons. The force carrier for the electromagnetic force is the photon, for the strong force it's the gluon and for the weak force it's the W^+, W^- and Z^o bosons. Currently, the fourth fundamental force, gravity, is not described within the Standard Model. To date, all known elementary particles and resonances can be described as bound states of quarks and gluons, with either two valence quarks for the mesons or three for the baryons.

In order to describe the dynamics of these fundamental particles, quantum theories are required. The dynamics of the electromagnetic interactions between charged particles is described by quantum electrodynamics (QED), of the electro-weak force (combination of electromagnetic and weak force) by the Weinberg-Salam model [4, 5] and of the strong force interactions between the quarks and the gluons by quantum chromodynamics (QCD). QCD is a non-Abelian Yang-Mills theory with a triplet of coloured quarks, its associated anti-triplet and an octet of coloured gluons [2]. This theory is non-Abelian since the force carriers (gluons) are themselves coloured, thus allowing self interaction of the gluons. This behaviour is not present within QED.

The form of the QCD potential results in to two very interesting phenomena within the theory. At large distances the potential increases linearly with distance and thus more and more energy is needed as two quarks are separated further apart. Eventually sufficient energy will have been added to the system, in separating a pair of quarks, so as to create a quark anti-quark pair. On the creation of this pair, the created quark forms a bound state with one of the original quarks as the anti-quark forms a bound state with the other. This phenomenon of the non-Abelian Yang-Mills, known as confinement [2], explains why a single quark can not be seen on its own. The other interesting behaviour, asymptotic freedom [6, 7, 8, 9], occurs at small distances or high energy densities. In this regime the strong force between two quarks tends to zero and the quarks behave essentially as free particles, at least with respect to the strong force.

A limitation in the study of the properties and interactions of the quarks and gluons has been the extraordinarily high energies required to do this. Suppose you wish to study a medium by probing it with a point-like particle, then the resolution to which the probing particle can measure is limited by its own de Broglie wavelength $\lambda = \frac{h}{p}$, where h is the Plank's constant and p is its momentum. In fact, the resolution is dependent on the momentum transfer between the probing particle and the objects in the target medium. Therefore, in order to probe inside a proton, which has radius of 1 fm, to a resolution of 0.1 fm would require a momentum transfer of approximately 10 GeV.

Accelerator	Location	Year built	Collision Type	Physics Discoveries
Proton Synchrotron (PS)	CERN, Switzerland	1959	Fixed target	neutral current interaction
Alternating Gradient Synchrotron (AGS)	BNL, USA	1960	Fixed target	muon neutrino and the J/ψ
Stanford Linear Accelerator Centre (SLAC)	Stanford, USA	1966	Fixed Target	charm quark, tau lepton and J/ψ
Super Proton Synchrotron (SPS)	CERN, Switzerland	1976	Fixed Target	W and Z bosons
Large Electron-Positron Collider (LEP)	CERN, Switzerland	1989	Ring Collider	Results imply that only three generations of quarks and leptons exist
The Hadron-Elektron-Ring-Anlage (HERA)	DESY, Germany	1992	Ring Collider	confirmed the nature of the strong force and the unification of the electromagnetic and weak forces
Tevatron	Fermilab, USA	1992	Fixed Target	top quark

Table 1.1 – Some of the particle accelerators used to study fundamental and elementary particles. The accelerators are listed with their location, the year they were built and some of the discoveries experiments held there led to.

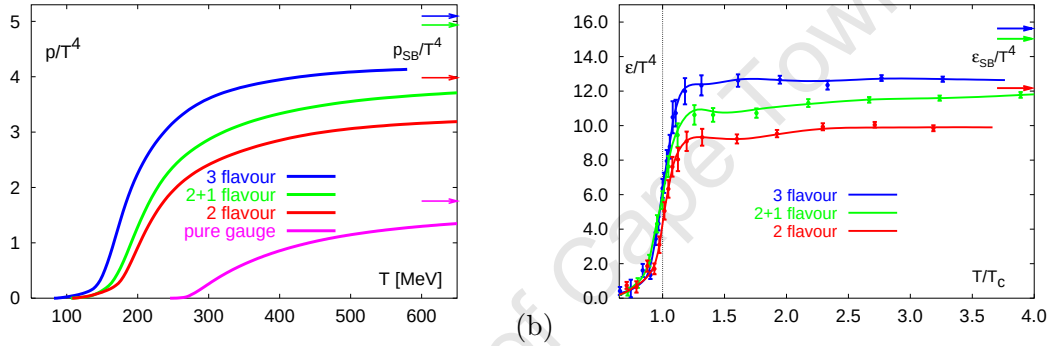
The Initial observations of elementary particles were only possible with the use of high energy cosmic rays. As technology improved the construction of particle accelerators was possible, providing a more controlled environment for research. Some of these accelerators are listed in Table 1.1. Over time certain colliders were converted to allow for the collision of nuclei, for example the AGS and SPS, and also used as injectors for larger accelerators like:

- The Relativistic Heavy Ion Collider (RHIC), Brookhaven National Laboratory (BNL), USA, which is an intersecting storage ring particle accelerator and Collider and collides Au-Au, Cu-Cu, deuterium-Au as well as polarised protons. The centre-of-mass energies attained per nucleon-nucleon collision range from 62.4 to 200 GeV. Results from RHIC have shown the first viable indications of the Quark Gluon Plasma.

- The Large Hadron Collider (LHC), Conseil Europen pour la Recherche Nuclaire (CERN), Switzerland, which is a ring accelerator and collides p-p and Pb-Pb. The LHC came on-line this year (2008).

1.1 Heavy Ion Collisions

At high energy densities/temperatures the coupling constant (α_s) for QCD is small, thus it is possible to make use of an effective perturbative theory (pQCD) by taking an expansion in α_s to study the dynamics. However, as the temperature decreases the coupling constant increases and eventually becomes too large for the validity of pQCD. Thus to understand the dynamics in this region one must make use of other methods, for example through lattice gauge theory calculations of QCD (lQCD). The results from lQCD calculations indicate the presence of a phase change as the temperature is varied, as is evident from Fig 1.1, with the sharp increase in the pressure (a) and energy density (b) indicating a change in the number of degrees of freedom. This can be explained as



(a) (b)

Figure 1.1 – Taken from [10]. The pressure (a) and energy density (b) in QCD with $n_f = 0$, 2 and 3 light quarks as well as two light and a heavier (strange) quark. Arrows indicate the ideal gas pressure p_{SB} (a) and energy densities ε_{SB} (b).

a change from hadronic degrees of freedom to the degrees of freedom associated with the individual quarks and gluons, *deconfinement*. lQCD calculations with 2 flavours indicate that a phase change occurs at around a critical temperature of T_c=173±8 MeV [11] while more recent calculations for (2+1) flavour QCD show a critical temperature of T_c=172±11±7MeV [12].All of these lQCD calculations were done in the vanishing net baryon density limit.

A QCD phase diagram is shown in Fig 1.2 as a function of temperature (T) and baryon chemical potential (μ_B), together with the asymptotic properties of QCD. For low μ_B the calculations from lQCD predict that the transition from the hadron phase to the QGP stage is a smooth crossover (dotted line) [13]. At higher μ_B it is expected that the transition is a first order phase transition and the critical point is the small region where a second order phase transition is expected.

If the partons in the deconfined system are also equilibrated, it is expected that a state of matter called the quark-gluon plasma (QGP) will exist. The asymptotic high energy limit of the QGP is expected to be a Stephen-Boltzmann (SB) gas of weakly

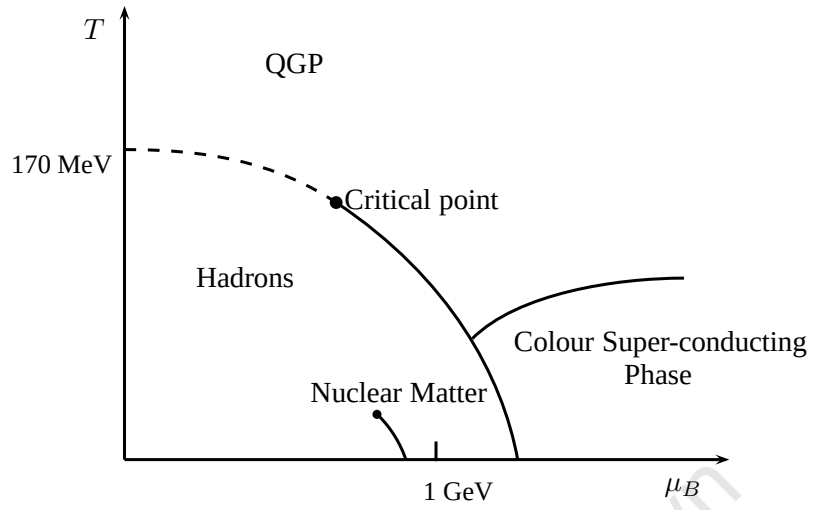


Figure 1.2 – The QCD phase diagram in terms of temperature (T) and baryon chemical potential (μ_B).

interacting particles and the limiting values for the energy density and pressure are shown as arrows in Fig 1.1. Gaining a better knowledge of the QGP is crucial in understanding matter and its fundamental building blocks. It is also important in understanding our universe as it is expected that such a QGP stage existed during the early stages of the Big Bang, before hadronisation [14]. The QGP is also expected to exist in the centre of neutron stars and its presence will have an affect on the spin down characteristics in the spectrum of rotating neutron stars [15].

Heavy ion collisions (HIC) were proposed as a way of constructing a system with a high enough temperature for the creation of the QGP [16]. During a heavy ion collision a region of matter which is extremely dense and energetic is produced. If the densities become sufficiently large, the quarks and gluons within this system should enjoy asymptotic freedom. Thus the individual hadrons are no longer distinguishable and the system becomes deconfined. Although temperatures above T_c are obtained at SPS [17, 18] there is only enough time for the partons to equilibrate at RHIC [19, 20] and the LHC.

1.2 The Collision

In the centre of mass frame of a relativistic HIC, the two colliding nuclei appear as disks due to Lorentz contraction. A diagram of the geometry of a collision is shown in Fig 1.3, with (i) a particular side view and (ii) the view down the beam line. The axis along which the two nuclei approach each other defines our z-axis, the beam axis of the HIC. The impact parameter, b, is the distance between the centres of the two nuclei at impact.

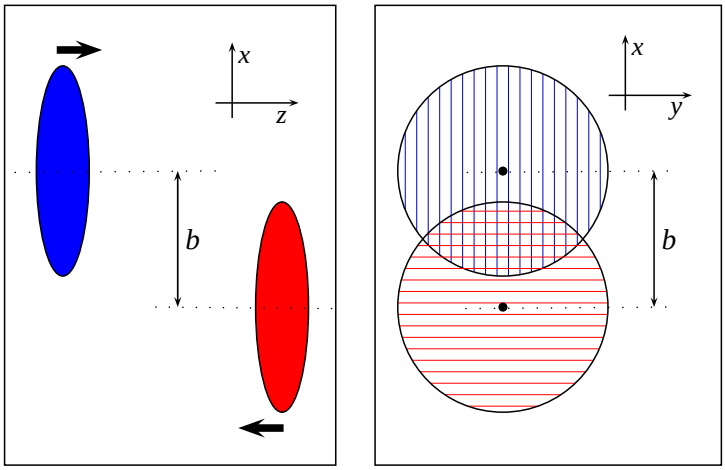


Figure 1.3 – Geometry of colliding nuclei in heavy ion collisions. (i) As viewed from perpendicular to the collision axis and the line through the centres of the nuclei (ii) As viewed down the collision axis

Thus a central collision will have $b = 0$. This impact parmeter also defines our x-axis and then the y-axis is chosen such that we have a right-handed coordinate system. The reaction plane is defined by the x- and z- axes, while the transverse plane is the plane defined by the x- and y-axes.

The Glauber model [21], a semi-classical description of the collision, can be used to determine the number of nucleons which participate in the collision. These participatory nucleons are termed the *wounded* nucleons [22, 23]. The Glauber model calculates the number of nucleons in the overlapping region in Fig 1.3(ii), assuming a Wood-Saxon' s distribution of nucleons in the two nuclei. The nucleons outside the collision area, which do not participate in the collision, are the *spectator* nucleons. Another parameter of interest in HICs is the number of binary collisions[24, 25] which occur between the constituent partons and can also be determined from the Glauber model. These parameters are relevant in HICs as at SPS energies the particle multiplicity yields scale with the wounded nucleons, while for central collisions at RHIC's higher energies they scale with binary collisions.

1.3 Evolution

If the energy densities created during the collision are sufficiently high, a system of deconfined quarks and gluons will form. These partons will then undergo rescattering and fragmentation as the system expands. The expansion in turn results in the decrease of the energy densities which eventually become too low for the deconfinement of the

partons and hadronisation occurs. A mixed phase state where hadrons and deconfined partons coexist should exist, due to the formation time of the hadrons. Eventually all the partons will be confined within hadrons resulting in a pure hadron gas. Such a hadron gas would also be created by HICs if the energy densities created by the collision were not sufficient for deconfinement. Thereafter, the hadrons in this gas rescatter inelastically until chemical freezeout, at which the particle numbers and ratios are fixed since the inelastic collisions have ceased. When the densities have decreased further and elastic scattering ceases, thermal freezeout has been reached and the momenta of the particles no longer change.

The simulation of the collision dynamics can be achieved with microscopic methods. For example, HIJING [26] can be used to determine the initial distributions and used as input for a parton cascade model, like VNI [27] or ZPC [28], to simulate the partonic stage. Then a hadronisation model is required to describe hadronisation, after which models like UrQMD [29] and ART [30] can simulate the evolution of the hadronic stage. There are some multi-phase models, such as AMPT [31], which have different componets to treat these separate stages combined into a single package. Using microscopic models requires tracking all the partons from the first collisions as well as all the secondary particles, their fragmentation and scattering with each other throughout the entire evolution. Unfortunately, for reasonable results, the microscopic descriptions of the partonic section require scattering cross sections which are orders of magnitude higher than those used by other partonic calculations like HIJING. There is also the difficulty of forming colourless observables during hadronisation.

However, if the system is thermalised, it can be described by a few thermodynamic macro variables rather than the multitude of micro variables. Moreover, if the scattering lengths are smaller than the size of the system then pressure gradients will drive the dynamic evolution of the macro variables. In this case, the dynamics of the system can be described by hydrodynamics; local conservation of energy, momentum and any conserved charges in the system. The system of dynamical equations provided by hydrodynamics is closed with the equation of state (EOS). This is a description of the pressure as a function of local energy density and possibly other local densities. Given an equation of state and an initial condition, a system which obeys hydrodynamic evolution is fully determined.

Much of the early applications [32] of hydrodynamics were due to the conceptual simplicity of hydrodynamics. When applied to AGS and SPS, hydrodynamics tends to overpredict the observables. However, the systems obtained at RHIC [33] and those to be obtained at the LHC have scattering lengths which are much smaller than the system size. In fact, right from the beginning, observables at RHIC displayed strong evidence of

collective flow [34, 35], motivating the use of hydrodynamics to explain the observables. Thus hydrodynamics has been widely used at RHIC [36, 37, 38, 39] and will be crucial in understanding the observables at the LHC.

1.3.1 Ideal Hydrodynamics

There are many excellent reviews on hydrodynamics as applied to heavy ion collisions [40, 41, 42, 43, 44] and introductory literature [45, 46, 47] to hydrodynamics. Here we present only a brief overview.

By making use of tensors, the local conservation of energy and momentum can be expressed elegantly as

$$\partial_\mu T^{\mu\nu} = 0, \tag{1.1}$$

where $T^{\mu\nu}$ is the energy-momentum tensor. Similarly the local conservation of conserved currents N^μ can be written as

$$\partial_\mu N^\mu = 0. \tag{1.2}$$

If in addition, the system is in local thermal equilibrium, then it can be described as an ideal fluid, which has no stresses nor viscosity. Thus the energy-momentum tensor is characterised purely by the local rest frame energy density (ε), the pressure (P) and the flow four-velocity (u^μ):

$$T^{\mu\nu} = (\varepsilon + P)u^\mu u^\nu + P g^{\mu\nu}, \tag{1.3}$$

where $g^{\mu\nu} = \text{diag}(1, -1, -1, -1)$ is the Minkowski metric. We are using natural units $\hbar = c = 1$. The flow four-velocity has the form $u^\mu = \frac{1}{\sqrt{1-v^2}}(1, v_x, v_y, v_z)$, with v_x,v_y and v_z the flow velocities in the x, y and z directions respectively and $v^2 = \sqrt{v_x^2 + v_y^2 + v_z^2}$. In this notation the local rest frame is defined as the frame in which $u^\mu = (1, 0, 0, 0)$. When a conserved charge is present in the system, the Eckart frame [48] is usually used. In this frame the flow four-velocity follows the conserved charge current. In the absence of a conserved charge one must make use of the Landau frame [49] were the flow four-velocity follows the energy-flow. For ideal fluids the two frames are identical.

The system of equations (1.1), for an ideal fluid, is a system of four independent equations, while there are five unknowns: energy, pressure and the three spatial components of the flow velocity (v_x, v_y, v_z). The final degree of freedom, as mentioned previously, is fixed through the equation of state (EOS).

1.3.2 Initial Conditions

Hydrodynamic simulations solve partial differential equations and as such require initial conditions for the distributions of the variables. These are the distributions at the point

where thermalisation has been achieved after the collision of the nuclei. Hydrodynamics does not care how thermalisation is achieved, it is up to other models to determine how thermalisation has been attained.

It is important to note that the final particle distributions are very sensitive to the initial pressure gradients [50] and are thus closely linked to the initial asymmetry of the system [51, 39]. At SPS it has been assumed that the initial entropy density [50, 52, 53] and alternatively that the initial energy density [36, 37, 38] scale with the wounded nucleon distribution. At RHIC, the binary collisions dominate the particle multiplicities and so the initial energy density has also been assumed to be proportional to the binary collision distribution [54, 55].

For the longitudinal direction, the simplifying assumption of longitudinal boost invariant scaling [56] is often made. Although this is the infinite collision energy limit, the transparency of the nuclei in ultra-relativistic heavy ion collisions makes it a viable assumption. Some models have relaxed this constraint and have obtained initial profiles which have been successful at reproducing observables at SPS [57, 58, 59] and RHIC [54, 59].

Other options include calculating the initial conditions by melting the colour glass condensate of the colliding nucleons [60], using string ropes/flux tubes [61] and using event generators such as HIJING [62], URASIMA [63] and NeXus [64].

1.3.3 Freeze-out

As the system expands it cools down and eventually a temperature is reached at which the mean free paths become too large for hydrodynamics to be applicable. Near this temperature it may be better to describe the particles as free streaming particles. To simulate the system accurately there should be a gradual change from hydrodynamics to free streaming.

In simulations, one simplification is to switch to a cascade model and treat the dilute hadronic gas microscopically. The general assumption is that this switch should occur shortly after hadronisation. The first work in combining hydrodynamics with a cascade model was done for cylindrically symmetric systems (central collisions) in [65, 66, 67] and non-cylindrically symmetric systems (peripheral collisions) in [68, 52].

Another simplification is to assume that the boundary between the region of hydrodynamics and that of free streaming is a hypersurface. We use the terminology that the particles have "frozen out through the freezeout surface". They are no longer the constituents of a fluid, but individual, unaffected particles. This freeze out scenario is

termed Cooper-Frye freezeout [69] and the number of particles frozen out is

$$N = \int_{\sigma} N^{\mu} \, \mathrm{d}\sigma_{\mu}, \tag{1.4}$$

where N^{μ} is the particle four-current, σ_{μ} is the freeze-out hypersurface and $\mathrm{d}\sigma_{\mu}$ is the outward normal to the freeze-out hypersurface. If the particles emitted from the freeze-out hypersurface have the distribution function $f(p^{\nu}, u^{\mu})$, then the particle four-flow is given by:

$$N^{\mu} = \int \frac{\mathrm{d}^3 p}{p^0} f(p^{\nu}, u^{\mu}). \tag{1.5}$$

This Cooper-Frye formalism has the unfortunate feature that it allows for negative contributions, due to particles "flowing back into the hydrodynamic region". If the expansion of the system is faster than the motion of a free particle it is possible, within this formalism, for the particle to re-enter the hydrodynamic system through the hypersurface. This feature should not occur as free particles should be free but the contributions are a few percent [52] and are usually ignored. Sometimes this feature is corrected for by introducing a Heavyside step function preventing negative contributions, unfortunately, this ad hoc addition violates energy-momentum conservation. There have been some other improvements on this formalism [70, 71, 72], however their implementation is complicated. A more recent move has been to extend the hypersurface to a hypervolume over which the particles freeze-out [73, 74].

1.3.4 Flow observables

In non-central collisions the collision region has a spatial anisotropy in the transverse plane and thus the initial pressure gradients have an azimuthal dependency. Since it is the pressure gradients which drive the dynamics, there will be azimuthal anisotropies in the momenta of the particles in the transverse plane. It is these azimuthal anisotropies which are referred to as the *flow observables*. These flow observables are effects of collective motion and hydrodynamics has been fantastic at reproducing them at RHIC at midrapidities and transverse momenta $p_T \leq 2$ GeV.

The azimuthal anisotropies can be expressed as a Fourier series [75]

$$E\frac{\mathrm{d}^3 N}{\mathrm{d}^3 p} = \frac{1}{2\pi}\frac{\mathrm{d}^2 N}{p_T \, \mathrm{d}p_T \, \mathrm{d}y}\left[1 + \sum_{n=1}^{\infty} 2v_n \cos(n(\phi - \Psi))\right], \tag{1.6}$$

where ϕ is the azimuthal angle, Ψ is the azimuthal angle of the reaction plane (the angle of the x-axis) and v_n are the Fourier coefficients. Collisions between identical nuclei result in a collision region symmetric about the reaction plane which results in the absence of

the sine terms in the expansion. It is the Fourier coefficients, v_n, which are used to analyse and quantify the flow of the systems. The coefficient v_1 is termed directed flow and v_2, elliptic flow.

At RHIC (Au-Au) the collision region for non-central collisions has an elliptic shape and the different pressure gradients, in the direction of the short versus the long axis, gives rise to the elliptic flow in hydrodynamic simulations. The elliptic spatial anisotropy of the initial condition is usually characterised by the eccentricity

$$\epsilon = \frac{\langle y^2 \rangle - \langle x^2 \rangle}{\langle y^2 \rangle + \langle x^2 \rangle}. \tag{1.7}$$

This is used to calculate the scaled elliptic flow $\frac{v_2}{\epsilon}$ which hydrodynamics predicts to be approximately constant at a value of 0.2 [37, 38, 39]. It was collisions at RHIC where the experimental value for the scaled elliptic flow first reached this limit. There have also been theoretical investigations and measurements of higher harmonic coefficients, such as v_4 [76, 77].

The deviation between theory and experiment for the elliptic flow at central rapidities occurs at a p_T of around 2 GeV [78]: The experimental data start to level off while the hydrodynamic prediction keeps increasing, see Fig 1.4. It is interesting to note that for

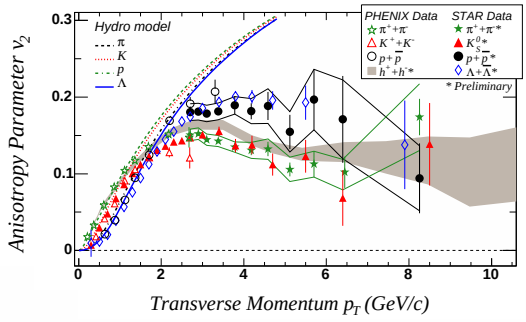


Figure 1.4 – Taken from [79]. Elliptic flow measurements at middle rapidity from minimum-bias Au+Au collisions at $\sqrt{s_{NN}} = 200$ GeV. The bands around the STAR preliminary measurements of pions and protons represent systematic uncertainties mostly from non-flow correlations. The PHOENIX measurements are made by correlating hadrons at middle rapidity with an event-plane measured using hadrons at $3.1 < \eta < 4.0$..

mesons and baryons the elliptic flow at RHIC for $2 \lesssim p_T \gtrsim 6$ GeV saturates to different values. However, if v_2 and the transverse momentum are both scaled by the number of valence quarks and plotted against each other (Fig 1.5) the lines converge. This suggests that the flow develops during the partonic stage of the evolution.

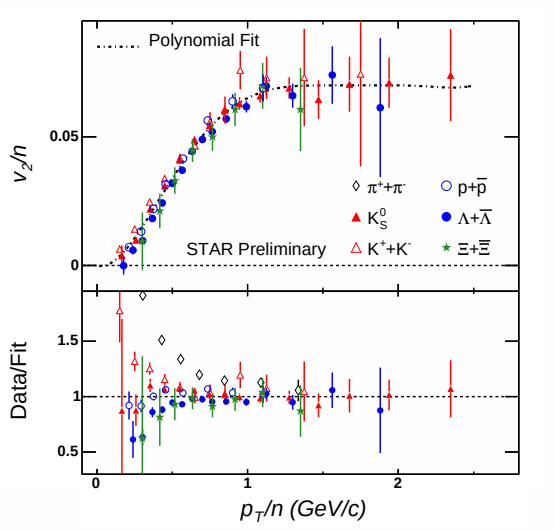


Figure 1.5 – Taken from [80]. Elliptic flow for different particle species scaled according to the number of constituent quarks of the hadrons. The lower plot shows the ratio of the data to the dashed-dotted fit to the data in the upper plot. Pions were not included in this fit and are only shown in the lower panel.

1.3.5 Nonideal Hydrodynamics

The application of ideal-hydrodynamics to the observables at RHIC only hold in certain regimes. As mentioned, the v_2 predictions are only valid for central rapidities ($|y| < 0.5$) and transverse momenta $p_T \leq 2$ GeV while at forward and backward rapidities ideal hydrodynamics also overpredicts the data. However, observables for forward and backward rapidities at RHIC have been described by introducing a thermalisation coefficient [81].

For some reason the system no longer behaves as an ideal fluid. Either the system is too dilute or perhaps internal friction and thermal conductivity are affecting the observables. If the system is not in thermal equilibrium then the thermodynamic irreversible processes (viscosity and thermal conductivity) lead to energy dissipation and the system can be described by nonideal hydrodynamics.

First order dissipative corrections to non-relativistic ideal hydrodynamics result in the well known Navier-Stokes equations and can be generalised to relativistic systems [48, 49]. However, these theories are acausal since the dynamic equations generated are parabolic. One must consider up to second order, dissipative corrections to obtain a causal theory. The second order non-relativistic system is given in [82] and was generalised to a relativistic system in [83]. It is important to note that there are other causal models that have also been developed [84, 85, 86].

There have been a number of applications of nonideal hydrodynamics to HIC, both using first order [87, 88], second order [89, 90] and other causal models [91, 92]. Recently

simulations have been done for non-central collisions [93, 94, 95]

1.4 ROOT

The data volumes which LHC will provide are orders of magnitude larger than previous experiments. For this reason, ROOT, an object orientated framework for large volume data analysis, was developed, and has become the primary framework for analysis of HIC events. Due to its prevalence in the HIC community there have been a number of applications (e.g. THERMUS for thermal model calculations, ...) which have been written to work within ROOT. In fact, it is such a useful framework that ROOT has been used to develop applications outside the area of physics, for example in the financial sector. However, as yet, there is no ROOT application for simulating hydrodynamic systems. It is important to introduce such an application since hydrodynamic effects have been shown to be important when regarding momenta observables. The main work of this thesis aims to address this issue. By introducing such an application, we hope to make hydrodynamic simulations more accessible to the broader HIC community and allow them to become more frequently used.

Chapter 2

FCTHydro

FCTHydro is a C++ package designed for the ROOT framework [96] for simulating hydrodynamical systems. The package can be loaded as a library in ROOT and its routines are then accessable from the command line in ROOT. As ROOT is the framework used by physicists in heavy-ion collisions, hopefully **FCTHydro** will broaden the use of hydrodynamic simulations in heavy ion collisions. This package is also designed such that the size of the system simulated is dynamic, in other words the size of the system can be set at runtime. Previousely (for example in Fortran programs) it was necessary to specify the size of the grid within the code itself. Thus each time the grid size was changed, the code required recompiling before it could be run. With a dynamic grid size, this is no longer necessary.

There are many different numerical schemes available for simulating a hydrodynamical system, for example Lax-Wendroff, leapfrog, donor cell (or upstreaming) and smooth particle. The algorithm we make use of is a flux corrected transport (FCT) algorithm which is designed for solving one-dimensional continuity equations. This FCT algorithm is especially useful when dealing with a system which contains large gradients. One tends to find that most other Eulerian convection methods introduce large numerical diffusion or else over correct and introduce numerical dispersion. Thus the FCT algorithm is ideal for describing hydrodynamic systems which contain shock waves, as the shock waves are accompanied by discontinuities and thus large gradients, in the densities, across the shock front.

The FCT algorithm handles these difficult conditions by applying a *corrected* diffusion to the dispersive transport. This is achieved by adding sufficient diffusion everywhere and then removing it where it is not needed via a corrected antidiffusion. This antidiffusion is nonlinear as it is determined locally for each point. Thus the diffusion is localised to regions where non physical ripples would normally occur due to dispersion. It is important to note that this diffusion-antidiffusion method is applied in a conservative

manner.

A flux corrected transport algorithm can be broken into three distinct stages. The first step is to convect or transport the density, then the density is diffused and finally there is the antidiffusion stage.

The algorithm which we make use of is the one employed in LCPFCT Fortran library [97]. It is a one-dimensional FCT algorithm with fourth-order phase accuracy designed for solving generalised continuity equations of the form

$$\frac{\partial \rho}{\partial t} + \frac{1}{r^{\alpha-1}}\frac{\partial}{\partial r}(r^{\alpha-1}\rho v) = \frac{1}{r^{\alpha-1}}\frac{\partial}{\partial r}(r^{\alpha-1}D_1) + C_2\frac{\partial D_2}{\partial r} + D_3, \tag{2.1}$$

which we have extended slightly to equations of the form

$$\frac{\partial \rho}{\partial t} + \frac{1}{r^{\alpha-1}}\frac{\partial}{\partial r}(r^{\alpha-1}\rho v) = C_1\frac{1}{r^{\alpha-1}}\frac{\partial}{\partial r}(r^{\alpha-1}D_1) + C_2\frac{\partial D_2}{\partial r} + D_3. \tag{2.2}$$

As one can see from the above equations, the algorithm is designed for Cartesian $\alpha = 1$, polar $\alpha = 2$ and spherical coordinates $\alpha = 3$.

This flux correct transport is a high-order, monotone, positivity preserving, conservative algorithm and as already stated, is accurate when dealing with steep gradients.

The well known SHASTA algorithm [98], which has been used in many phenomenological simulations of heavy-ion collisions, is also a flux corrected transport system. One difference between these algorithms is that the values at the cell interfaces are used in LCPFCT as opposed to the cell centre values as in SHASTA.

2.1 The FCT Algorithm

We will now review the particular FCT algorithm [97] to be used. During this discussion the following notation convention is adopted for consistency: values defined at cell centres are indicated with integer subscripts i while those with subscripts $i + \frac{1}{2}$ describe values at the cell interface between cells i and $i + 1$. The values at the interface $i + \frac{1}{2}$ are the averages of the values at the cell centres i and $i + 1$. Similarly $i - \frac{1}{2}$ describes the values at the interface between cells i and $i - 1$.

The algorithm is also generalised in such a manner that both Eulerian and Lagrangian grids* can be implemented. The interfaces between cells are the finite-difference grid points, where, as described before, $r^{o,n}_{i+\frac{1}{2}} = \frac{1}{2}(r^{o,n}_i + r^{o,n}{}_{i+1})$. The values of the grid

*The spatial positions of the cells in a Lagrangian grid can change from timestep to timestep, while in a Eulerian grid system they are fixed.

points before (r^o) and after (r^n) the timestep Δt are related by

$$r^n_{i+\frac{1}{2}} = r^o_{i+\frac{1}{2}} + v^g_{i+\frac{1}{2}}\Delta t,$$

where v^g is the grid velocity. Obviously, for a Eulerian grid $v^g = 0$.

Since the algorithm depends on the fluxes into and out of the cells we need the difference between the fluid velocity, v_f, and the grid velocity, which we define as

$$\Delta v_{i+\frac{1}{2}} = v^f_{i+\frac{1}{2}} - v^g_{i-\frac{1}{2}}. \tag{2.3}$$

We also require the densities at the cell interfaces at the beginning of the timestep,

$$\rho^o_{i+\frac{1}{2}} = \frac{1}{2}(\rho^o_i + \rho_{i+1}). \tag{2.4}$$

The values of these parameters at the end points of the finite-difference grid, $i = \frac{1}{2},\ N+\frac{1}{2}$, are determined by applying the appropriate boundary conditions via setting the cell values at $i = 0,\ N+1$. There are four different types of boundary conditions one can apply at the moment; periodic boundary conditions, anti-periodic, a hard wall boundary condition and the last is a user specified boundary condition which is set through the equations

$$\rho_0 = S_1\rho_1 + V_1, \tag{2.5}$$

$$\rho_{N+1} = S_N\rho_N + V_N. \tag{2.6}$$

The values of S_1, S_N are the user set slope multipliers and V_1 and V_N are user set additive values.

Fig 2.1 gives an indication of the geometry of the system for the one-dimensional flow. As stated previously the $r_{i+\frac{1}{2}}$ are the finite-difference grid points and are the interfaces

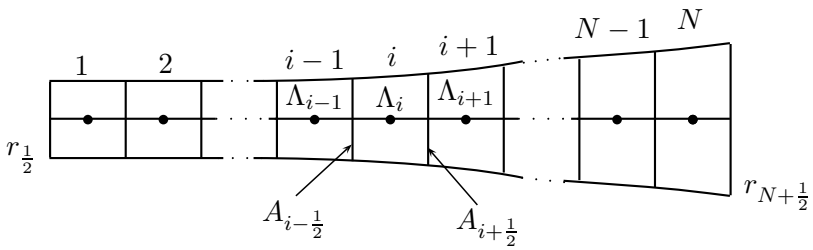


Figure 2.1 − Geometry of one-dimensional grid

between cells i and $i+1$. The $A_{i+\frac{1}{2}}$ represent the area of the interface between cells i and $i+1$, while Λ_i is the volume of cell i. For the three different coordinate systems

mentioned above, these are define as

$$\Lambda_i^{o,n} = \begin{cases} \left[r_{i+\frac{1}{2}}^{o,n} - r_{i-\frac{1}{2}}^{o,n} \right] & \text{Cartesian} \\ \pi \left[(r_{i+\frac{1}{2}}^{o,n})^2 - (r_{i-\frac{1}{2}}^{o,n})^2 \right] & \text{Cylindrical} \\ \frac{4}{3}\pi \left[(r_{i+\frac{1}{2}}^{o,n})^3 - (r_{i-\frac{1}{2}}^{o,n})^3 \right] & \text{Spherical} \end{cases} \tag{2.7}$$

and

$$A_{i+\frac{1}{2}} = \begin{cases} 1 & \text{Cartesian} \\ \pi \left[r_{i+\frac{1}{2}}^{o} + r_{i+\frac{1}{2}}^{n} \right] & \text{Cylindrical} \\ \pi \left[(r_{i+\frac{1}{2}}^{o})^2 + (r_{i+\frac{1}{2}}^{n})^2 + r_{i+\frac{1}{2}}^{o} r_{i+\frac{1}{2}}^{n} \right] & \text{Spherical} \end{cases} \tag{2.8}$$

2.1.1 Convection

With these definitions we can now convect the density via

$$\Lambda_i^o \rho_i^* = \Lambda_i^o \rho_i^o - \Delta t \left[\rho_{i+\frac{1}{2}}^o A_{i+\frac{1}{2}} \Delta v_{i+\frac{1}{2}} - \rho_{i-\frac{1}{2}}^o A_{i-\frac{1}{2}} \Delta v_{i-\frac{1}{2}} \right]. \tag{2.9}$$

What we now have is the purely convected density multiplied with the old cell volume as the size of the cells have not yet changed. The next step is to add the sources, i.e. the contribution due to the terms on the right hand side of (2.2). This then results in the convected and transported density

$$\begin{aligned} \Lambda_i^o \rho_i^T = \Lambda_i^o \rho_i^* &+ \frac{1}{2}\Delta t A_{i+\frac{1}{2}} C_{1,i} \left[D_{1,i+1} - D_{1,i} \right] - \frac{1}{2}\Delta t A_{i-\frac{1}{2}} C_{1,i} \left[D_{1,i} - D_{1,i-1} \right] \\ &+ \frac{1}{4}\Delta t C_{2,i} \left(A_{i+\frac{1}{2}} + A_{i-\frac{1}{2}} \right) \left[D_{2,i+1} - D_{2,i-1} \right] + \Delta t \Lambda_i^o D_{3,i} \end{aligned} \tag{2.10}$$

Note: The boundary values of D_1, D_2, D_3, C_1 and C_2 must be set by the user.

2.1.2 Diffusion

We can now add the diffusion, yielding the transported and diffused density $\tilde{\rho}_i$;

$$\Lambda_i^n \tilde{\rho}_i = \Lambda_i^o \rho_i^T + \nu_{i+\frac{1}{2}} \Lambda_{i+\frac{1}{2}} \left[\rho_{i+1}^o - \rho_i^o \right] - \nu_{i-\frac{1}{2}} \Lambda_{i-\frac{1}{2}} \left[\rho_i^o - \rho_{i-1}^o \right] \tag{2.11}$$

The left hand side of this equation contains the volume of the cell at the end of the time step Δt as this stage incorporates the changes due to the possible expansion or contraction of size of the cells, in a Lagrangian grid system. The $\Lambda_{i+\frac{1}{2}}$ are the averaged values of Λ_i^n and Λ_{i+1}^n. By choosing the values of the diffusion coefficients $\nu_{i+\frac{1}{2}}$ and the antidiffusion coefficients $\mu_{i+\frac{1}{2}}$ appropriately, it is possible to reduce the phase errors from [97] second order to fourth order. This is a correction which most SHASTA algorithms do not employ, and as such generally have second order phase errors.

2.1.3 Antidiffusion

Now we have to calculate the antidiffusive fluxes, correct them and apply them to the transported and diffused density $\tilde{\rho}_i$. There are three available choices; explicit, implicit and phonetical antidiffusion [99]. To understand the difference between these options we will assume that we are considering a Eulerian grid, that there are no sources and adopt the following notation [99]: The operation of transporting (or convecting) the density will be denoted by the operator T, diffusion by D and antidiffusion by A. With these, the different antidiffusion schemes can be expressed as

$$\rho_E^n = (1+A)(1+T+D)\rho^0, \tag{2.12}$$

$$\rho_I^n = (1+D)^{-1}(1+T+D)\rho^0, \tag{2.13}$$

$$\rho_P^n = [(1+A)(1+T)+D]\rho^0. \tag{2.14}$$

The problem with the explicit antidiffusion is that it has a residual diffusion which damps short wavelengths, even for vanishing flow $v \to 0$ ($T \to 0$). The implicit scheme removes this residual diffusion [98], however it can give erroneous results near shocks. Secondly it depends on a numerically costly recurrence relation.

The antidiffusion we make use of is the phonetical antidiffusion which introduces higher order terms to the antidiffusive fluxes which mean that the diffusion is exactly cancelled when the flow vanishes $v \to 0$. This scheme derives its name from "phoenix": 'features which sagged into shapelessness in the diffusion stage are "reincarnated" and resorted to their original form during antidiffusion'[99].

For this choice we need the transported but not diffused densities ρ_i^T to calculate the antidiffusive flux

$$f_{i+\frac{1}{2}}^{ad} = \mu_{i+\frac{1}{2}} A_{i+\frac{1}{2}} \left[\rho_{i+1}^T - \rho_i^T\right]. \tag{2.15}$$

This is also designed such that when the grid is Lagrangian and Δv vanishes we have $\Lambda_i^n \rho_i^n = \Lambda_i^o \rho_i^o$.

It was shown in [100] that to reduce the order of the phase errors on a locally uniform

grid to forth order, one should make the choices:

$$\nu_{i+\frac{1}{2}} = \frac{1}{6} + \frac{1}{3}\epsilon^2_{i+\frac{1}{2}}, \tag{2.16}$$

$$\mu_{i+\frac{1}{2}} = \frac{1}{6} - \frac{1}{6}\epsilon^2_{i+\frac{1}{2}}, \tag{2.17}$$

for the diffusion constants, where

$$\epsilon_{i+\frac{1}{2}} = \frac{1}{2}A_{i+\frac{1}{2}}\Delta v_{i+\frac{1}{2}}\Delta t\left[\frac{1}{\Lambda^n_i} + \frac{1}{\Lambda^n_{i+1}}\right]. \tag{2.18}$$

If the antidiffusive fluxes calculated above are applied, there is the risk of creating maxima and minima in the antidiffusive stage which causes the numerics to become unstable. Thus the antidiffusive fluxes at each point need to be corrected. This correction is done according to the values of the density at that point and the surrounding points. The corrected antidiffusive fluxes are calculated using the algorithm

$$f^c_{i+\frac{1}{2}} = S_{i+\frac{1}{2}}\max\left\{0, \min\left\{\left|f^{ad}_{i+\frac{1}{2}}\right|, S_{i+\frac{1}{2}}\Lambda^n_{i+1}\left[\tilde{\rho}_{i+2} - \tilde{\rho}_{i+1}\right], S_{i+\frac{1}{2}}\Lambda^n_i\left[\tilde{\rho}_i - \tilde{\rho}_{i-1}\right]\right\}\right\} \tag{2.19}$$

with $S_{i+\frac{1}{2}}$ the sign of $[\tilde{\rho}_{i+1} - \tilde{\rho}_i]$. This ensures that if $\tilde{\rho}_i$ is below $\tilde{\rho}_{i+1}$ then the antidiffusive flux does not push $\tilde{\rho}_i$'s value above that of $\tilde{\rho}_{i+1}$.

To understand what occurs when we correct the fluxes we must realise that $f^c_{i+\frac{1}{2}}$ affects both $\tilde{\rho}_i$ and $\tilde{\rho}_{i+1}$ in the antidiffusive stage. Consider the situation where $S_{i+\frac{1}{2}} > 0$. The four possible situations as shown in Fig 2.2. If the density profile around a specific

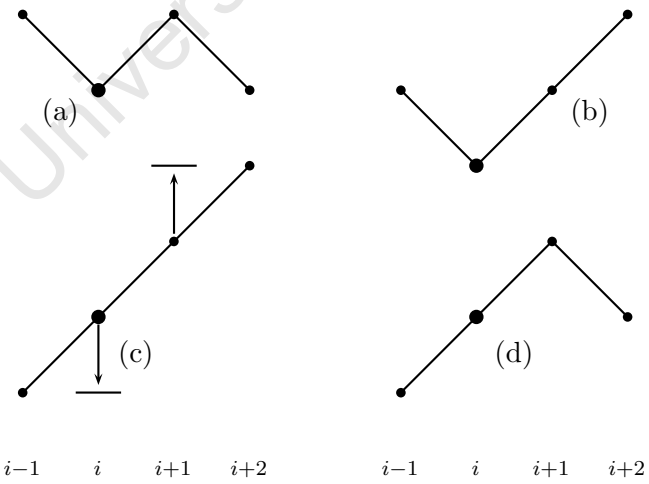


Figure 2.2 − Calculation of the corrected antidiffusive flux

point i (as indicated by the larger point) follows the behaviour as show in Fig 2.2(a,b,d) then the antidiffusive flux is set to zero. This prevents the antidiffusive fluxes from

enhancing extrema.

However, if the profile follows the behaviour described in Fig 2.2(c) the the flux is left at its calculated value as long as it does not move the $\tilde{\rho}_i$ below $\tilde{\rho}_{i-1}$ and also as long as it does not move $\tilde{\rho_{i+1}}$ above $\tilde{\rho_{i+2}}$. If either of these would occur the antidiffusive flux is corrected to avoid this. The overall affect is to steepen the profile when the antidiffusive stage is applied. This type of correction is called strong flux correction and can tend to leave a large net diffusion. Another consequence of this correction is a property called "clipping". Clipping occurs because of the property that no extrema which are present after the diffusion stage are allowed to be enhanced by the antidiffusion. Thus a sharp peak profile will be blunted [99]. There are possible alternatives but they all require knowledge of the character of the solution.

Now that the antidiffusive fluxes have been corrected we can calculate the density values at the end of the time step:

$$\rho_i^n = \tilde{\rho}_i - \frac{1}{\Lambda_i^n}\left[f^c_{i+\frac{1}{2}} - f^c_{i-\frac{1}{2}}\right] \tag{2.20}$$

2.1.4 Two Step

The algorithm can be used with a two step integration method method, where for each time step the densities are evolved forward half a time step $\frac{\Delta t}{2}$, using the values for the sources and velocity profile at the beginning of the time step. The source terms and velocity are then recalculated using the density at this stage and used to evolve the densities from the beginning of the time step forward a full time step Δt. The two step method can be described by the following schematic [97]:

- Start with density, velocity and sources at the beginning of the time step: ρ^0, v^0 and D_j^0.
- Evolve density ρ^0 forward half a time step $\frac{\Delta t}{2}$ using v^0 and D_j^0 to $\rho^{\frac{1}{2}}$.
- Recalculate velocity $v^{\frac{1}{2}}$ and sources $D_j^{\frac{1}{2}}$ using $\rho^{\frac{1}{2}}$.
- Evolve density ρ^0 forward a full time step Δt using the half timestep velocity and source values; $v^{\frac{1}{2}}$ and $D_j^{\frac{1}{2}}$.

This method, reduces the numerical errors dramatically.

2.1.5 Multidimensional systems

The flux corrected algorithm described above is a one dimensional algorithm. However most physical systems of interest are multidimensional. This one-dimensional algorithm can be used for multidimensional systems through the timestep split method.

For this method the system is evolved in each of the dimensional directions using the one-dimensional algorithm for each time step

This method is illustrated through the following example of a two-dimensional non-relativistic hydrodynamic system [97]. The continuity equations describing the dynamics of this system are

$$\begin{aligned} \partial_t \rho + \partial_x(\rho v_x) + \partial_y(\rho v_y) &= 0, \\ \partial_t(\rho v_x) + \partial_x(\rho v_x^2) + \partial_y(\rho v_x v_y) &= -\partial_x P, \\ \partial_t(\rho v_y) + \partial_x(\rho v_y v_x) + \partial_y(\rho v_y^2) &= -\partial_y P, \\ \partial_t E + \partial_x(E v_x) + \partial_y(E v_y) &= -\partial_x(P v_x) - \partial_y(P v_y). \end{aligned} \tag{2.21}$$

To apply the timestep slip method we evolve the system in the x-direction for each fixed value of y, then in the y-direction for each fixed value of x. This is done for each timestep Δt. The continuity equations for the evolution in the x-direction are

$$\begin{aligned} \partial_t \rho + \partial_x(\rho v_x) &= 0, \\ \partial_t(\rho v_x) + \partial_x(\rho v_x^2) &= -\partial_x P, \\ \partial_t(\rho v_y) + \partial_x(\rho v_y v_x) &= 0, \\ \partial_t E + \partial_x(E v_x) &= -\partial_x(P v_x), \end{aligned} \tag{2.22}$$

and those for the y-direction are

$$\begin{aligned} \partial_t \rho + \partial_y(\rho v_y) &= 0, \\ \partial_t(\rho v_x) + \partial_y(\rho v_x v_y) &= 0, \\ \partial_t(\rho v_y) + \partial_y(\rho v_y^2) &= -\partial_y P, \\ \partial_t E + \partial_y(E v_y) &= -\partial_y(P v_y). \end{aligned} \tag{2.23}$$

It is important to keep the timesteps small enough so that the fluxes do not change the value of the cell values too much when evolving in either of the directions. When using the timestep split method it is advisable to alternate the order of evolution directions, so as to avoid introducing a bias into the system.

This method is easily generalised to three-dimensional systems.

2.1.6 Non-Conservative Convection

There is also a modification to this FCT algorithm for solving non-conservative convection equations of the form

$$\frac{\partial p}{\partial t} + v\frac{1}{r^{\alpha-1}}\frac{\partial}{\partial r}(r^{\alpha-1}\rho) = C_1\frac{1}{r^{\alpha-1}}\frac{\partial}{\partial r}(r^{\alpha-1}D_1) + C_2\frac{\partial D_2}{\partial r} + D_3. \tag{2.24}$$

Note the difference between this equation and the conservative convection equation (2.2): in this instance the velocity appears outside the gradient term, on the left hand side of the equation. The only change to the algorithm occurs in the convection stage, where (2.9) is replace by:

$$\Lambda_i^o\rho_i^* = \Lambda_i^n\left[\rho_i^o - \frac{2\Delta t\Delta v_{i+\frac{1}{2}}(\rho_{i+1}^o - \rho_i^o)}{r_{i+\frac{3}{2}}^n - r_{i-\frac{1}{2}}^n + r_{i+\frac{3}{2}}^o - r_{i-\frac{1}{2}}^o} + \frac{2\Delta t\Delta v_{i-\frac{1}{2}}(\rho_i^o - \rho_{i-1}^o)}{r_{i+\frac{1}{2}}^n - r_{i-\frac{3}{2}}^n + r_{i+\frac{1}{2}}^o - r_{i-\frac{3}{2}}^o}\right]. \tag{2.25}$$

Now that the transported density ρ^* is calculated, the sources, diffusion and antidiffusion are calculated in exactly the same manner as above.

Table 2.1 – Table of variables used in the FCT algorithm

Δt	time for one timestep
r_i^o	position of cell centres at beginning of timestep
r_i^n	position of cell centres at end of timestep
v^f	velocity of fluid
v^g	velocity of grid (Lagrangian)
Λ_i	Volume of cell
$A_{i+\frac{1}{2}}$	Area of cell interface
ρ_i^o	density at beginning of timestep
ρ_i^*	convected density
ρ_i^T	transported density
$\tilde{\rho}_i$	diffused density
ρ_i^n	density at end of timestep
$f_{i+\frac{1}{2}}$	antidiffusion flux
$f_{i+\frac{1}{2}}^c$	corrected antidiffusion flux
$\nu_{i+\frac{1}{2}}$	diffusion coefficient
$\mu_{i+\frac{1}{2}}$	antidiffusion coefficient

2.2 The FCTHydro Code

FCTHydro consits of various classes which are compiled into a shared library for loading and running in the ROOT environment [96]. This will hopefully broaden the use of

hydrodynamic evolution simulations in heavy-ion collisions. It consits of the base classes **FCTFluid**, **FCTVeloctiy**, **FCTGrid** and **FCTVector**.

We will now describe the **FCTVector**, **FCTGrid**, **FCTVelocity** and **FCTFluid** classes. For each we will review the important members and member functions. The members are listed with the name of the member followed by the data type of the member and then a description of the member. The functions are listed with the name of the function with the input variables and is followed by the output type. The use of each function is also provided. At the end of each section there is a diagram indicating the order in which the functions must be called. An arrow means that the function(s) to the left of the arrow must be called before the function(s) to the right of the arrow. Any input variable with an "s" prefix indicates it is used to set the value of a member of the class.

The members of the classes which correspond to parameters of the flux corrected transport algorithm with integer subscripts, i.e. the cell centre values, have N+2 components. The first $(i = 0)^{\dagger}$ and last elements $(i = N + 1)$ are set according to the boundary conditions. The values of $i = 1...$N are the values for the physical cell centres
The members corresponding to the cell interface values, i.e. those in the algorithm with half integer superscripts have N+1 elements. The first element $i = 0$ is for the lower boundary cell interface $r_{\frac{1}{2}}$ and $i =$ N is for the upper boundary cell interface $r_{N+\frac{1}{2}}$.

2.2.1 FCTVector

This is a class for a dynamic array of doubles. It is this class which gives **FCTHydro** the dynamic system size functionality. As stated before, the benefit of this functionality is that the size of grid for the system you wish to simulate can be decided at run time instead of before compile time. **FCTVector** contains the following members

N	Int_t	size of the vector
vector	Double_t []	pointer to dynamically set vector of doubles

and the following important member functions

SetSize(Int_t sN)	void
Fill(Double_t $fillvalue$)	void
operator[ ](Int_t N)	Double_t&
operator=(const FCTVector& $sVec$)	FCTVector&

[†]Since the system is coded in C++ the first component an array has index $i = 0$.

As the name suggests, the SetSize() sets the size of the array of doubles to the value of sN. Fill() is used to fill the entire array with the value fillvalue. The operator "[ ]" takes the input integer N and returns the value of the array at position N. The overloaded operator "=" sets the array of the **FCTVector** object to the same size and the same entries as that of $sVec$.

The order dependence of the member functions:

$$\text{SetSize()} \rightarrow \begin{cases} \text{Fill()} \\ \\ \text{operator[ ]} \end{cases}$$

2.2.2 FCTGrid

This class contains all the geometrical values which depend on the grid cells and the functions for initialising the grid and calculating these geometrical values. The following members are important.

N	Int_t	number of cell centres
$alpha$	Int_t	coordinate systems (2.7), (2.8)
dx	Double_t	used for initialising a uniform grid
$x0$	Double_t	the lower boundary of the grid
$oldvolElement$	FCTVector	old volume of cell; $oldvolElement[i] = \Lambda_i^0$
$volElement$	FCTVector	volume of cell; $volElement[i] = \Lambda_i^n$
$hArea$	FCTVector	$hArea[i] = A_{i+\frac{1}{2}}$
x	FCTVector	cell centres $x[i] = r_i$
hx	FCTVector	cell interface positions; $hx[i] = r^n_{i+\frac{1}{2}}$
$oldhx$	FCTVector	old cell interface positions; $oldhx[i] = r^o_{i+\frac{1}{2}}$
$hgridVel$	FCTVector	cell interface velocities; $hgridVel[i] = V^g_{i+\frac{1}{2}}$

Recall that the superscript "o" indicates the value at the beginning of the timestep and "n" the value at the end. Now we list a few of the important member functions;

SetParams(Int_t sN, Int_t $salpha$, Double_t sdx, Double_t $sx0$)	void
SetVectorSize()	void
SetDeltat(Double_t vdt)	void
SetPeriodicBCs(Bool_t $spBCs$)	void
SetGridVelocity()	void
InitialiseGrid()	void
TranslateGrid()	void

As is indicated by the name, SetParams() is used to set the values of the members N, $alpha$, dx and $x0$. SetVectorSize() sets the size of the vectors. This is done as described above; the parameters with integer subscripts have N+2 elements to account for the boundary values, and the parameters with half-integer subscripts have N+1 elements. Once the variables are set with SetParams() and SetVectorSize() has been called, InitialiseGrid() can be called. This calculates the initial values of the geometrical parameters of the grid. TranslateGrid() is called to calculate the new geometrical parameters (values at the end of the timestep). This function must be called at least once, even if you are using a Eulerian grid, as $volElement$ is calculated for the first time here. If you are using periodic boundary conditions for this grid direction, then SetPeriodicBCs() must be called with **true**. SetDeltat() has been created to allow for easily varying the step size of the timesteps. (section §2.1.4).

The order dependence of the member functions:

$$\left.\begin{array}{r}\text{SetParams()} \rightarrow \text{SetVectorSize()} \\ \text{SetPeriodicBCs()}\end{array}\right\} \rightarrow \text{InitialiseGrid()} \left.\begin{array}{r}\\ \text{SetGridVelocity()} \\ \text{SetDeltat()}\end{array}\right\} \rightarrow \text{TranslateGrid()}$$

2.2.3 FCTVeloctiy

This class contains all of the velocity dependent parameters and the necessary functions for calculating them. Some important members of this class are

$maxN$	Int_t	size of arrays allocated in memory
N	Int_t	size of the system being simulated
$BC1$	Int_t	type of boundary condition at lower edge
BCn	Int_t	type of boundary condition at upper edge
$scrhSwitch$	Double_t	variable for adjusting diffusion and antidiffusion coefficients
$fluidVel$	Double_t [ ]	flow velocity at cell centre; $fluidVel[i] = v_i^f$
$hfluidVel$	Double_t [ ]	flow velocity at cell interface; $fluidVel[i] = v_{i+\frac{1}{2}}^f$
$hdeltaFluid$	Double_t [ ]	difference between flow velocity and grid velocity at cell interface; $hdeltaFluid[i] = \Delta v_{i+\frac{1}{2}}$
nu	Double_t [ ]	velocity dependent diffusion coefficient; $nu[i] = \nu_{i+\frac{1}{2}}$

mu	Double_t [ ]	velocity dependent antidiffusion coefficient; $mu[i] = \mu_{i+\frac{1}{2}}$

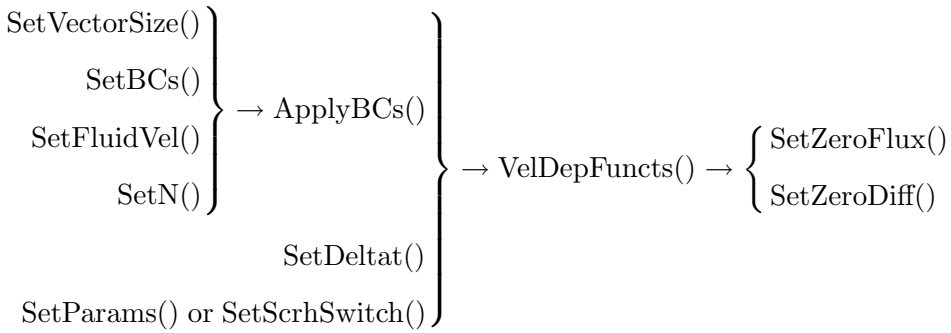
Having both N and $maxN$ allows one to have only one **FCTVeloctiy** object when working with a multidimensional system. In this case $maxN$ is set to the size of the largest grid dimension. Then N is set to the size of the grid for the dimension in which you are evolving (when using the split timestep method section §2.1.5). See section §4 for an application of this functionality. If the is grid uniform, $scrhSwitch = 0$ and the velocity is constant, then the simulation has fourth-order phase accuracy. However, better results for simulations can be obtained by adjusting $scrhSwitch$, but the choice for optimisation will depend on the system simulated.
The following member functions are useful

SetParams(ifstream $*confile$)	TString
SetVectorSize (Int_t $smaxN$)	void
SetBCs(Int_t $sBC1$, Int_t $sBCn$, Double_t $sS1 = 1$, Double_t $sV1 = 0$ Double_t $sSn = 1$, Double_t $sVn = 0$)	void
SetFluidVel(Double_t $*valFluidVel$)	void
SetDeltat(Double_t vdt)	void
SetN(Int_t sN)	void
SetZeroFlux(Int_t i)	void
SetZeroDiff(Int_t i)	void
SetScrhSwitch(Double_t $sscrhSw$)	void
ApplyBCs()	void
VelDepFuncts(**FCTGrid** $*Grid$)	void

The function SetParams() takes an pointer to a configuration file which specifies the value for $scrhSwitch$. Else SetScrhSwitch() must be called to set its value. SetBCs() is used for setting the boundary conditions at the lower and upper ends of the grid. The user set values for ($sS1$, $sV1$) and (sSn, sVn) are only used if $BC1 = 1$ or $BCn = 1$, respectively. The function ApplyBCs() applies the boundary conditions to the $fluidVel$. The function SetN() is used for setting N, while SetVectorSize() allocates the memory of the arrays according to $maxN$. SetDeltat() performs the same function as in the **FCTGrid** class.

VelDepFuncts() takes as an argument the **FCTGrid** object corresponding to the direction in which the evolution is occurring and calculates all the velocity dependent parameters. SetZeroFlux() is used to ensure that there is no flux through a hard-wall boundary condition (BC=2) at interface $r_{i+\frac{1}{2}}$, while SetZeroDiff() is used at the boundaries to remove diffusion through a boundary (BC=1) at interface $r_{i+\frac{1}{2}}$.

The order dependence of the member functions:



2.2.4 FCTFluid

This class contains the necessary functions and parameters for convecting a specific density. Important members are

$maxN$	Int_t	size of arrays allocated in memory
N	Int_t	size of the system being simulated
$BC1$	Int_t	boundary condition at lower edge
BCn	Int_t	boundary condition at upper edge
$correctAntidiffFlux$	Int_t	bool switch for correcting antidiffusive fluxes
$residualDiff$	Double_t	parameter for adding extra diffusion to the system should be corrected; equation (2.19)
$hanitdiffFlux$	**FCTVector**	antidiffusive fluxes
$fluidMass$	**FCTVector**	density multiplied by cell volume
$density$	**FCTVector**	density values
$olddensity$	**FCTVector**	density values at the beginning of the timestep

The parameters $maxN$ and N perform the same function as in the **FCTVeloctiy** class. The important member functions are

SetParams(ifstream $*confile$)	TString
SetVectorSize (Int_t $smaxN$)	void
SetBCs(Int_t $sBC1$, Int_t $sBCn$, Double_t $sS1 = 1$, Double_t $sV1 = 0$ Double_t $sSn = 1$, Double_t $sVn = 0$)	void
SetSourceModes(Bool_t $sMode1$,Bool_t $sMode2$,Bool_t $sMode3$,	void

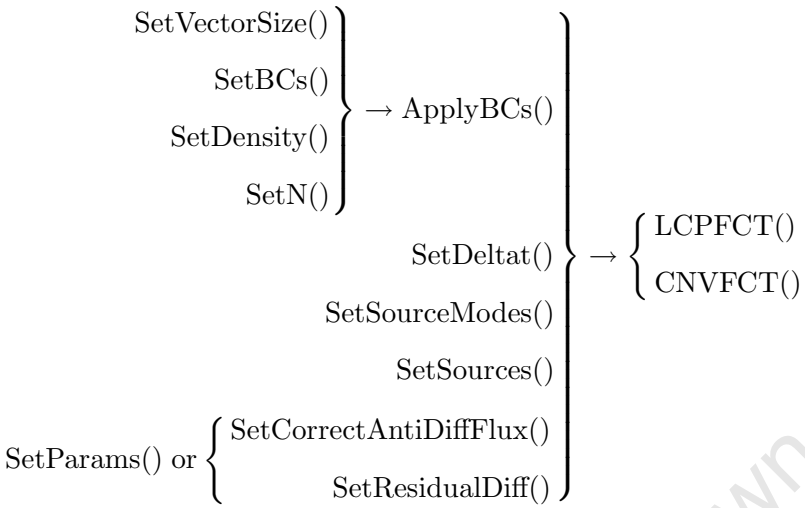
Bool_t $sMode4$)	
SetSources(Double_t *C1, Double_t *D1, Double_t *C2, Double_t *D2, Double_t *D3)	void
SetDensity(Double_t $*SDensity$)	void
SetDeltat(Double_t vdt)	void
SetN(Int_t sN)	void
SetCorrectAntiDiffFlux(Int_t $sCADF$)	void
SetResidualDiff(Double_t $sresdiff$)	void
ApplyBCs()	void
ApplyBCsTo(Double_t *sVector)	void
ConvectLCP(**FCTGrid** *Grid, **FCTVeloctiy** *Velocity)	void
ConvectCNV(**FCTGrid** *Grid, **FCTVeloctiy** *Velocity)	void
Sources(**FCTGrid** *Grid, **FCTVeloctiy** *Velocity)	void
Diffuse(**FCTGrid** *Grid, **FCTVeloctiy** *Velocity)	void
AntiDiffuse(**FCTGrid** *Grid, **FCTVeloctiy** *Velocity)	void
LCPFCT(**FCTGrid** *Grid, **FCTVeloctiy** *Velocity)	void
CNVFCT(**FCTGrid** *Grid, **FCTVeloctiy** *Velocity)	void

SetParams() take the pointer to a configuration file to set the values of $correctAntidiffFlux$ and $residualDiff$. If these are not set in the file then SetCorrectAntiDiffFlux() and/or SetResidualDiff() must be called. SetVectorSize(), SetN(), SetDeltat() and SetBCs() have the same function as they do in the class **FCTVelocity**. ApplyBCs() applies the boundary conditions to the *density* vector, while ApplyBCsTo() applies the boundary conditions to an array with pointer $sVector$ and is used to apply boundary conditions to source terms.

The function ConvectLCP() and ConvectCNV() perform the connective part of the evolution for the conservative equation (2.2) and the nonconservative equation (2.24), respectively. Sources() adds the contribution to due to the source terms (2.10), Diffuse() performs the diffusive part (2.11) and AntiDiffuse() calculates the corrected antidiffusive fluxes and performs the antidiffusive stage of the calculation. All of these functions take a pointer to an **FCTGrid** object and an **FCTVeloctiy** object as these objects hold the geometrical and velocity dependent parameters for the dimension in which the evolution is occurring.

Rather than call each of these functions separately one can call LCPFCT() to do a complete evolution for timestep size set by SetDeltat() for the equation (2.2). CNVFCT() does the same but for the nonconservative equation (2.24)

The order dependence of the member functions:



2.2.5 libFCTHydro

The above classes can be compiled into a shared library **libFCTHydro.so**, which can then be loaded into the ROOT environment. Thus one can write ROOT macros which use the **FCTHydro** to solve any number of generalised transport equations of the form of (2.2) or (2.24), in any number of dimensions. Another alternative is to write a class or set of classes to do this. This second option allows one to compile an executable which can be run without opening ROOT.

Thus any hydrodynamic simulation, from (1+1)-dimensional to (3+1)-dimensional, ideal, non-ideal, relativistic or non-relativistic can be run within ROOT. What changes from system to system is the manner in which **FCTHydro** is applied. In the next two chapters we present two such applications.

2.2.6 How To Use

A ROOT macro is provided below to illustrate how one can use **FCTHydro** to convect a distribution. This specific macro is used to convect a square wave with a constant velocity $v = 1$, according to equation (2.2) with no source terms.

```
gSystem->Load("~/Projects/FCTHydro/lib/libFCTHydro.so");
Int_t N=50;
Double_t dx=1.0;
Double_t dt=0.2;
Double_t x0=dx/2.0;
```

```
TRHFluid rhos(N);
TRHGrid grid(N,1,dx,x0);
TRHVelocity v(N);

rhos.SetBCs(2,2);
v.SetBCs(2,2);
rhos.SetSourceModes(0,0,0,0,0);
rhos.SetN(N);
v.SetN(N);
rhos.SetDeltat(dt);
v.SetDeltat(dt);

grid.InitialseGrid();
grid.TranslateGrid();

for (int i=1; i<=N; i++)
{
if( grid.x[i]<=10.0 ) {rhos.density[i] = 1.0;}
v.fluidVel[i] = 1.0;
}
v.ApplyBCs();
v.VelDepFuncs(&grid);

for(Int_t j=0;j<20;j++)
{
rhos.LCPFCT(&grid,&v);
}
```


The macro first loads the **FCTHydro** library. It then instantiates a **FCTFluid**, **FCTGrid** and a **FCTVeloctiy** object for a system with a maximum of N points. The grid is a cartesian grid ($\alpha = 1$) with grid spacing dx and its left edge at $x0$. The macro also sets the boundary conditions for a hard wall at both ends of the grid, turns off all sources, and sets the spatial size of the system and the timestep size. It then initialises the grid and translates it once to calculate necessary grid parameters. The fluid and velocity profiles are set, the velocity's boundary conditions applied and the velocity dependent variables calculated. These only need to be calculated once as the velocity is constant. Finally the density profile is convected forward conservatively 20 timesteps. In order to convect it non-conservatively one must replace the command "rhos.LCPFCT(&grid, &v)" by "rhos.CNVFCT(&grid, &v)".

2.3 Testing

To test whether the base classes of **FCTHydro** have been coded correctly we compared some simple simulations with those using LCPFCT Fortran routines [97]. These consist of a few convections of waves with different profiles and grids of different geometries. All

the tests were done on a Eulerian grid with $N = 50$ points, the left boundary at $x0 = 0$, grid spacing of $dx = 1.0$, timestep size $dt = 0.2$. A constant velocity of $v = 1.0$ was used and there are no sources terms.

The first three tests were done using periodic boundary conditions on a rectangular coordinate system ($\alpha = 1$), where three profiles, one a square wave, another a circular wave and the third a Gaussian, where convected across the grid. After 500 timesteps the output from **FCTHydro** was compared to that from the LCPFCT Fourtran routines and the exact solution. The Figures 2.3, 2.4 and 2.5 show the solution for the square wave, circular wave and the Gaussian distributions respectively. In each of the figures the **FCTHydro** solution is indicated by the solid line, the solution from the LCPFCT Fortran routines by the squares and the exact solution by the dotted line. If one plots the solutions from **FCTHydro** and the LCPFCT Fortran routines using lines, then they are indistinguishable to the eye for all three plots. There is a small difference, most likely due to the significant figures used when the output from the LCPFCT Fortran routines is saved to file. Another indicator of the accuracy of the code is the test

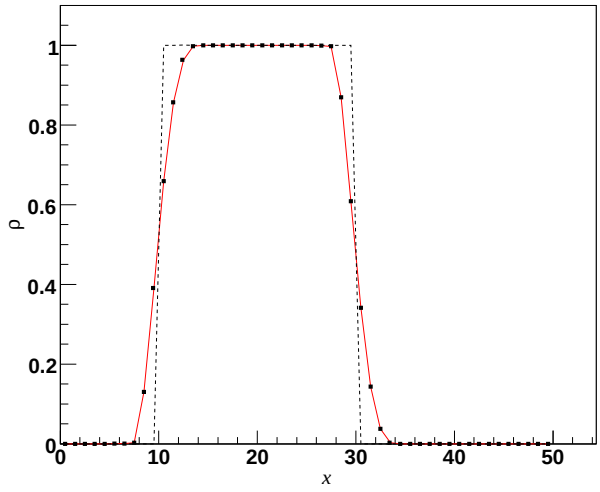


Figure 2.3 – Convection of a square wave distribution . The output from **FCTHydro** (solid line), LCPFCT Fortran routines (squares) and the exact solution (dotted line) are compared.

for conservation. The integral of the densities, over the grid, should be constant as the conservative convective transport (2.2), without source terms, was used. In Table 2.2 the difference in the integral of the densities before and after the evolution are compared. As is evident, the algorithm is incredibly accurate as there is only a deviation of the order of 10^{-15} after 500 timesteps.

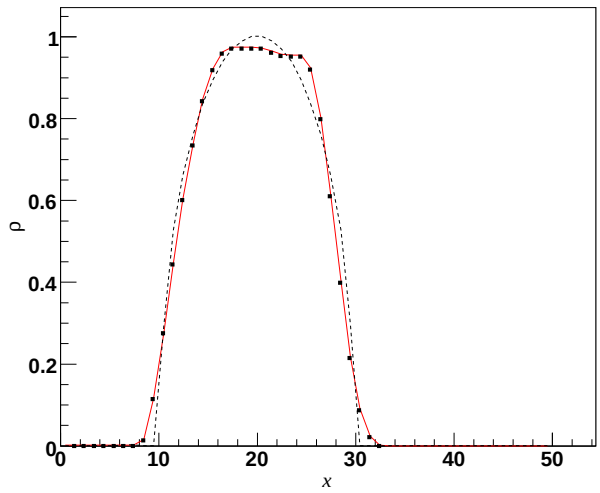


Figure 2.4 – Convection of a half circular wave distribution. The output from **FCTHydro** (solid line), LCPFCT Fortran routines (squares) and the exact solution (dotted line) are compared.

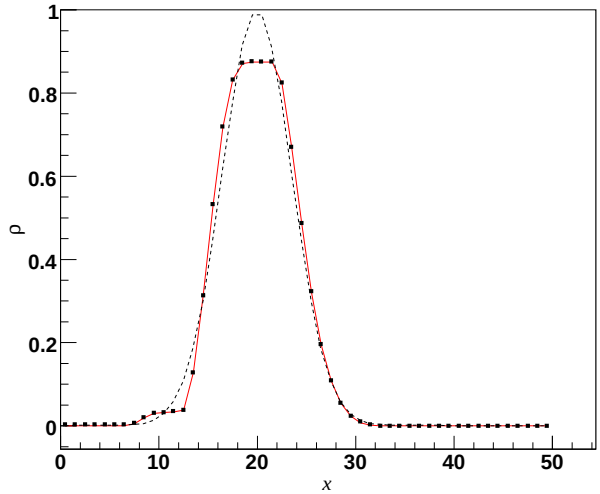


Figure 2.5 – Convection of a Gaussian distribution. The output from **FCTHydro** (solid line), LCPFCT Fortran routines (squares) and the exact solution (dotted line) are compared.

When the above convection tests are repeated, this time using the non-conservative convection equation (2.24) we obtain same results as for a constant velocity profile, equations (2.2) and (2.24) are equivalent.

Scenario	Initial value	Difference between initial and final
Square Wave	20	0
Circular Wave	1.5709682e1	5e-15
Gaussian	8.862262	5e-15
Polar Coordinates	2.5132741e3	5e-12
Spherical Coordinates	1.089085e5	1e-10

Table 2.2 – Conservation check: The integral of the initial densities is given as well as the difference between this and the integral of the final density from FCTHydro

The next tests were done using the polar ($\alpha = 2$) and spherical ($\alpha = 3$) coordinate system. The results in Fig 2.6 were obtained by transporting an initial square wave distribution according to the conservative transport equation (2.2) on a polar coordinate system for 75 timesteps. The same is done for the nonconservative convection equation (2.24) which is shown in Fig 2.7. As before the **FCTHydro** solution and the LCPFCT Fortran routines' solutions are indistinguishable to the eye.

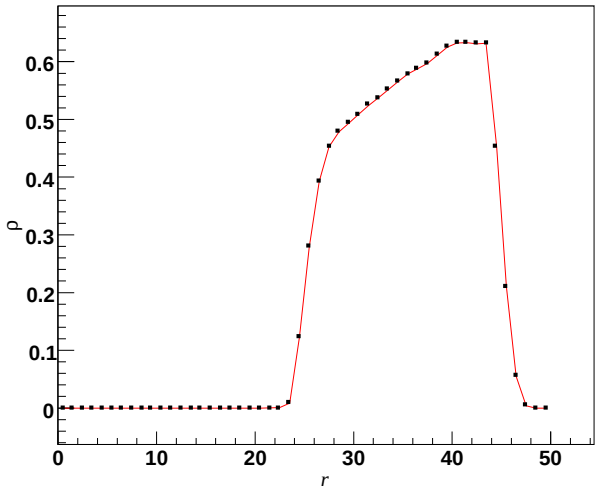


Figure 2.6 – Conservative convection of an initial square wave distribution on a polar coordinate system. The output from **FCTHydro** (solid line) and LCPFCT Fortran routines (squares) are compared.

The last two tests were done for a spherical coordinate system ($\alpha = 3$). Again, an initial square wave distribution is convected using the conservative equation, Fig 2.8, and non-conservative equation, Fig 2.9. And, again, the solutions are indistinguishable.

For both the polar and spherical coordinate systems the integral of the densities was calculated before and after the evolution, for the conservative evolutions. The results in

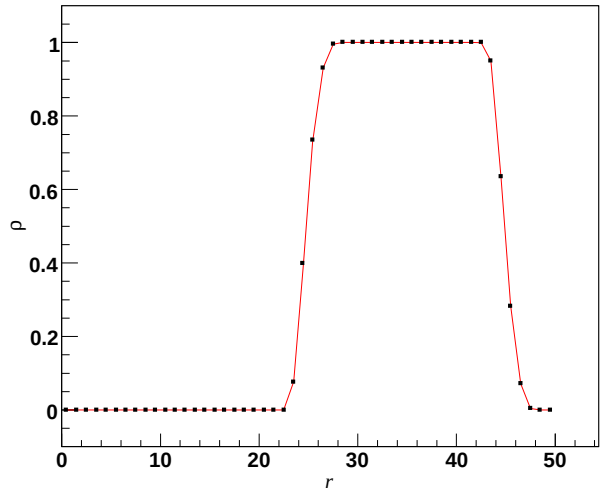


Figure 2.7 – Non-conservative convection of an initial square wave distribution on a polar coordinate system. The output from **FCTHydro** (solid line) and LCPFCT Fortran routines (squares) are compared.

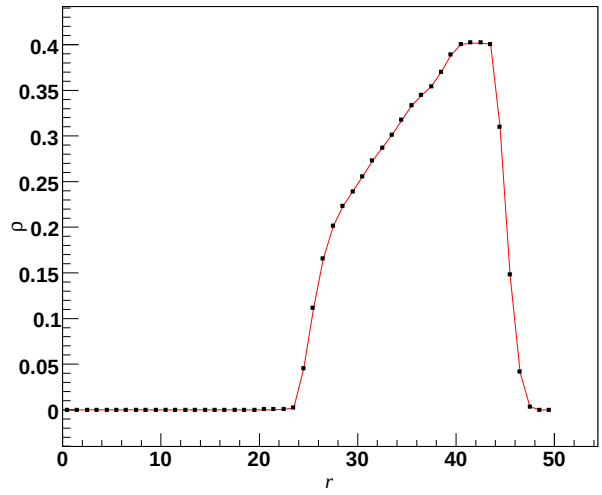


Figure 2.8 – Conservative convection of an initial square wave distribution on a spherical coordinate system. The output from **FCTHydro** (solid line) and LCPFCT Fortran routines (squares) are compared.

Table 2.2 show that the numerics are still very accurate, with a difference of the order of 10^{-12} and 10^{-10} for polar and spherical, respectively. The relative difference for all simulations performed is of the order of 10^{-15}.

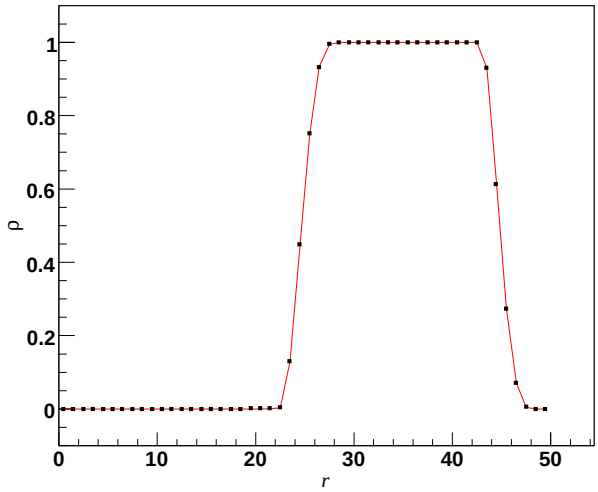


Figure 2.9 – Non-conservative convection of an initial square wave distribution on a spherical coordinate system. The output from **FCTHydro** (solid line) and LCPFCT Fortran routines (squares) are compared.

As mentioned, in the next two chapters we present two hydrodynamic applications of **FCTHydro**. The first is for a $(1 + 1)$ dimensional, relativistic, ideal hydrodynamic problem; **TRelHydro**. The second is for a (2+1) dimensional, non-ideal, non-relativistic application; **TNonideal_xy**. These applications are coded as classes with their necessary functions and compiled into share library files which can be loaded into ROOT, thus making thier functions available from the ROOT command line.

Chapter 3

TRelHydro - 1+1D

Now that **FCTHydro** provides tools for solving continuity equations, how can these tools be used to simulate hydrodynamic systems? More specifically, relativistic systems, as these are the ones of interest in heavy ion collisions. In answer, this chapter presents an application of **FCTHydro** to a (1+1) dimensional, relativistic, ideal hydrodynamic system. This is achieved via the package **TRelHydro**. By considering a simpler (1+1) dimensional system, it is easier to illustrate the application of **FCTHydro** to relativistic systems. Also the known analytic solutions to certain (1+1) dimensional systems can be used as a test for the accuracy of the numerics. **TRelHydro** is still a useful tool in its own right as (1+1) dimensional simulations are of great use in heavy ion collisions for obtaining first order approximations to the momenta distributions. In the next chapter other challenges are considered in the form of multi-dimensional, non-ideal hydrodynamic systems.

The first section of this chapter presents the theory behind the code in **TRelHydro**. The evolution equations for the (1+1) dimensional relativistic system are described as well as the various equations of state which are included in **TRelHydro**. There is also a description of the freeze-out process implemented. In the second section the code is presented in the same manner as for **FCTHydro**; a description of all classes along with their most important members and functions are provided. At the end of this section there is also a brief explanation on how to run a simulation using **TRelHydro**. In the final section **TRelHydro** is tested by applying **TRelHydro** to systems with known analytic solutions.

3.1 Theory

In (1+1) dimensions the flow four-velocity reduces to $u^\mu = \gamma(1, v, 0, 0)$, with $\gamma = \frac{1}{\sqrt{1-v^2}}$, and the hydrodynamic equations,

$$\partial_\nu T^{\mu\nu} = 0,$$

reduce to the equations:

$$\partial_t T^{00} + \partial_x T^{0x} = 0,$$
$$\partial_t T^{x0} + \partial_x T^{xx} = 0,$$

for the components of the energy-momentum tensor. Furthermore, for ideal hydrodynamic systems, equation (1.3), we can use the relations, $T^{0x} = (T^{00} + P)v$ and $T_{xx} = vT_{0x} + P$, to rewrite these equations as decoupled equations for T^{00} and T^{0x};

$$\begin{aligned} \partial_t T^{00} + \partial_x (vT^{00}) &= -\partial_x (vP), \\ \partial_t T^{0x} + \partial_x (vT^{0x}) &= -\partial_x (P). \end{aligned} \tag{3.1}$$

Similarly the equation for the conserved currents

$$\partial_\nu N^\nu = \partial_t(\gamma n) + \partial_x(\gamma v n) = 0,$$

can be written as

$$\partial_t(\tilde{N}) + \partial_x(v\tilde{N}) = 0, \tag{3.2}$$

where $\tilde{N} = \gamma n$ is the lab-frame density. The equations (3.1) and (3.2) are now in the form of the generalised conservative convection equation (2.2) solveble by **FCTHydro**. From the comparison with equation (2.2) we see that in (1+1) dimensional, ideal hydrodynamics we can evolve the components T^{00} and T^{0x}, of the energy-momentum tensor, with source terms $-\partial_x(vP)$ and $-\partial_x(P)$, respectively. While, conserved currents have no source terms and it is the lab-frame densities $\tilde{N}$ which are evolved.

The system of equations (3.1) and (3.2) are not fully deterministic as there are three equations and essentially four unknowns; the energy density, ε, pressure, P, fluid velocity, v and conserved number density, n[†]. Thus an equation of state is needed to supply the fourth constraint via an equation for the pressure as a function of the energy and number densities. To simulate this system the following formalism is followed

[†]In heavy ion collisions the baryon number is conserved but the particle number is not.

- Given a distribution for T^{00}, T^{0x} and $\tilde{N}$ at the beginning of a timestep, perform a fit using the equation of state to find the associated distributions of P and v.
- Use P and v to determine the values of T^{00}, T^{0x} and $\tilde{N}$ at the end of the timestep.

This method is repeated for as many timesteps as necessary.

Non-relativistic systems are easier to simulate since it is the energy, the various momentum components and the number density themselves which are evolved. Usually the equation of state can be rewritten as a fairly simple algebraic equation in terms of the evolved variables. As described above for relativistic hydrodynamics, though, it's the energy-momentum tensor components T^{00} and T^{0x} and the lab-frame density $\tilde{N}$ which are evolved, while the equation of state is in terms of local rest-frame densities. Thus it is very unusual to find algebraic equations for v and P in terms of T^{00}, T^{0x} and $\tilde{N}$, using the equation of state. Thus in relativistic hydrodynamics there is usually an extra step of computation needed at each timestep in terms of a numercial fit to find the values of P and v. If the equation of state has the simple form $P = c_s^2\varepsilon - \beta$, however, we can substitute it into equation (1.3) and obtain

$$\begin{aligned} \varepsilon &= \frac{1}{2c_s^2}\left[\sqrt{[(1+c_s^2)T^{00}-\beta]^2 - 4c_s^2(T^{0x})^2} + (c_s^2-1)T^{00} + \beta\right], \qquad (3.3)\\ P &= c_s^2\varepsilon - \beta,\\ v &= \frac{T^{0x}}{T^{00}+P}, \end{aligned}$$

For relativistic systems with this equation of state the square of, the speed of sound $\dfrac{\partial P}{\partial \varepsilon} = c_s^2$.

3.1.1 Equation of State

There are currently four different equations of state which have been coded and included in **TRelHydro**. One is for a gas of massless particles and another for a thermal hadron gas. The third is for a baryonless gas of quarks and gluons which undergoes a phase change to a gas of pions. Finally there is also the facility for a tabulated equation of state.

Massless Particles

This equation of state is for a gas of massless particles which obey Boltzmann, Fermi-Dirac or Bose-Einstein statistics. In fact, the gas can consist of various fractions of these types of particles. To calculate the equation of state for gases with these statistics, the

logarithm of the partition functions is required. In the case of Boltzmann particles it is given by

$$\ln Z = \frac{gTV}{(2\pi)^3} \int \mathrm{d}^3 p \, e^{\frac{\mu - E}{T}}, \tag{3.4}$$

while

$$\ln Z = \pm \frac{gTV}{(2\pi)^3} \int \mathrm{d}^3 p \ln \left(1 \mp e^{\frac{\mu - E}{T}}\right), \tag{3.5}$$

for Bose-Einstein (upper sign) and Fermi-Dirac (lower sign) particles, in terms of the temperature, T, the chemical potential, μ, the volume, V, the energy, E, and the degeneracy of the paticles, g. From these we are able to calculate the pressure, energy and particle density via the thermodynamic relations

$$P = \frac{\partial \ln Z}{\partial V}, \qquad \varepsilon = -\frac{1}{V} \frac{\partial \ln Z}{\partial (1/T)}, \qquad n = \frac{1}{V} \frac{\partial \ln Z}{\partial \mu}.$$

Table 3.1 gives these variables as a function of temperature for systems with zero net baryon density, i.e. systems where $\mu = 0$. From these results it clearly follows that the

Statistics	P	ε	n
Boltzmann	$\frac{g}{\pi^2} T^4$	$\frac{3g}{\pi^2} T^4$	$\frac{g}{\pi^2} T^3$
Bose-Einstein	$\frac{7g\pi^2}{720} T^4$	$\frac{7g\pi^2}{240} T^4$	$\frac{3g}{4\pi^2} \zeta(3) T^3$
Fermi-Dirac	$\frac{g\pi^2}{90} T^4$	$\frac{g\pi^2}{30} T^4$	$\frac{g}{\pi^2} \zeta(3) T^3$

Table 3.1 – The pressure, energy density and particle density of a gas of massless particles which obey Boltzmann, Bose-Einstein and Fermi-Dirac statistics are given for the case where $\mu = 0$.

equation of state, for all three statistics, is simply $p = \frac{\varepsilon}{3}$. Hence we can use the algebraic equations (3.3) with $c_s^2 = \frac{1}{3}$ and $\beta = 0$.

Thermal Hadrons

This is the equation of state for a thermal gas of hadrons including resonances. The code interfaces with the THERMUS package [101] to calculate the energy density, pressure and baryon density, in the grand-canonical ensemble. In this ensemble the logarithm, of the partition function for a multi-component hadron gas, is given by

$$\ln Z^{GC}(T, V, \mu_i) = \sum_{species\ i} \frac{g_i V}{(2\pi)^3} \int \mathrm{d}^3 p \ln \left(1 \pm e^{-\frac{E_i - \mu_i}{T}}\right)^{\mp 1} \tag{3.6}$$

with the upper signs for fermions, the lower sign for bosons and where μ_i is the particle chemical potential. For each individual particle i, its chemical potential is written in terms of its baryon B_i, strangeness S_i and charge number Q_i as well as the corresponding chemical potentials μ_B, μ_S and μ_Q, respectively:

$$\mu_i = B_i\mu_B + S_i\mu_S + Q_i\mu_Q. \tag{3.7}$$

In the current application of this equation of state, the charge density and strangeness density are zero. This is achieved by setting μ_Q to zero, while μ_S is constrained, such that the strangeness is zero. THERMUS also allows for a γ_S factor to account for possible incomplete strangeness saturation [102, 103]. Our default is to have $\gamma_S = 1$.

Since THERMUS gives the pressure, energy and baryon density, given the temperature and baryon chemical potential, a fitting function is used in conjunction with THERMUS to find the values of the temperature, μ_B and velocity for given values of T^{00}, T^{0x} and the lab-frame baryon density $\tilde{N}$. Using the calculated values of temperature and μ_B, the pressure, local rest-frame energy and baryon densities are calculated. This is done for each cell on the grid during each evolution timestep. This process can become incredibly resource consuming, and was the orginal reason for introducing the tabulated equation of state functionality.

Quarks, Gluons and Pions

This is the equation of state for a baryonless gas which undergoes a first order phase transition from a state of massless quarks and gluons to one of pions at some critical temperature. For the higher temperatures the gas of quarks and gluons has energy density and pressure:

$$\varepsilon_Q = \frac{\pi^2}{15}\left[N_c^2 - 1 + \frac{7}{4}N_cN_f\right]T^4 + B, \tag{3.8}$$

$$P_Q = \frac{1}{3}\varepsilon_Q - \frac{4}{3}B, \tag{3.9}$$

where N_c is the number of colours, N_f the number of flavours and B is the Bag constant. This is the equation of state for a Stephan-Boltzmann gas with a bag constant introduced and is the equation of state for the well know bag model [104]. The bag model is obtained by introducing a Bag constant term, $Bg^{\mu\nu}$, into the energy-momentum tensor, which moves the perturbative vacuum of the quarks and gluons below the level of the non-perturbative vacuum. Thus the confinement of quarks and gluons is described.

At lower temperature the gas of massless pions has

$$\varepsilon_\pi = \frac{\pi^2}{10} T^4, \tag{3.10}$$

$$P_\pi = \frac{1}{3}\varepsilon_\pi. \tag{3.11}$$

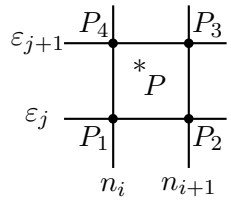
During the phase change the system is in a mixed phase of the above two gases and the temperature and pressure are constant. At this critical temperature, $T = T_c$, the pressure of the quark-gluon gas is the same as that of the pion gas $P_Q = P_\pi$. These criteria can be used in two manners, firstly one could fix the bag constant, B, and find the corresponding critical temperature. Alternatively, one could set the value of the critical temperature and find the corresponding value for the bag constant. Thus for energy densities above $\varepsilon_\pi\big|_{T_c}$ and below $\varepsilon_Q\big|_{T_c}$ the gas is in the mixed phase and $T = T_c$ and $P_Q = P_\pi = P_c$.

Since there is such a simple relation between the pressure and the energy density, equation (3.3) can be used with $c_s^2 = 1/3$ and $\beta = \frac{4}{3}B$ and for the quark-gluon gas and $c_s^2 = 1/3$ and $\beta = 0$ for the pion gas.

Tabulated EOS

This is an equation of state where the pressure is given in a tabulated form, as a function of the local rest-frame energy and baryon density. Thus we have a two dimensional grid with one axis corresponding to energy density, the other to baryon density and at each point in the grid the pressure is given.

To calculate the values of the pressure at points (n, ε), which do not correspond to grid points, we use a bilinear interpolation. In the diagram below, the corner points correspond to grid points, i.e. points where the pressure is known.



We would now like to calculate the pressure P at the point of interest, which corresponds to the point (n, ε); $n_i \leq n \leq n_{i+1}$, $\varepsilon_j \leq \varepsilon \leq \varepsilon_{j+1}$. With this setup the pressures at the

corner points are

$$P_1 = P[i][j] \qquad P_2 = P[i+1][j]$$
$$P_3 = P[i+1][j+1] \qquad P_4 = P[i][j+1],$$

where the first component corresponds to the baryon density and the second to the energy density. By defining the variables

$$t = \frac{n - n[i]}{n[i+1] - n[i]} \tag{3.12}$$

$$u = \frac{\varepsilon - \varepsilon[j]}{\varepsilon[j+1] - \varepsilon[j]} \tag{3.13}$$

we have, from bilinear interpolation, that P is given by

$$P = (1-t)(1-u)P_1 + t(1-u)P_2 + tuP_3 + (1-t)uP_4. \tag{3.14}$$

When this tabulated equation of state was used for the thermal hadron gas, an issues arose as there is a region in the (n, ε) plane of non-existence, as indicated in Fig 3.1. Thus

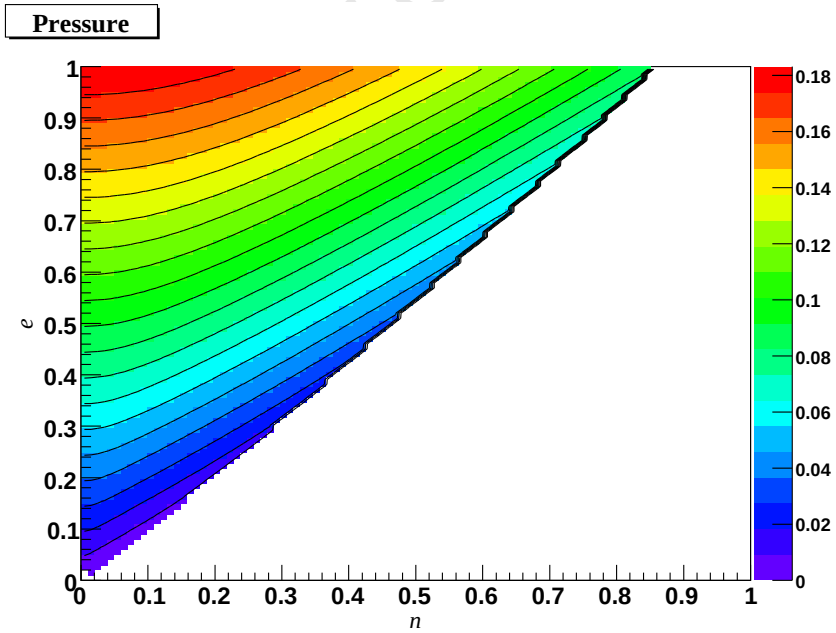
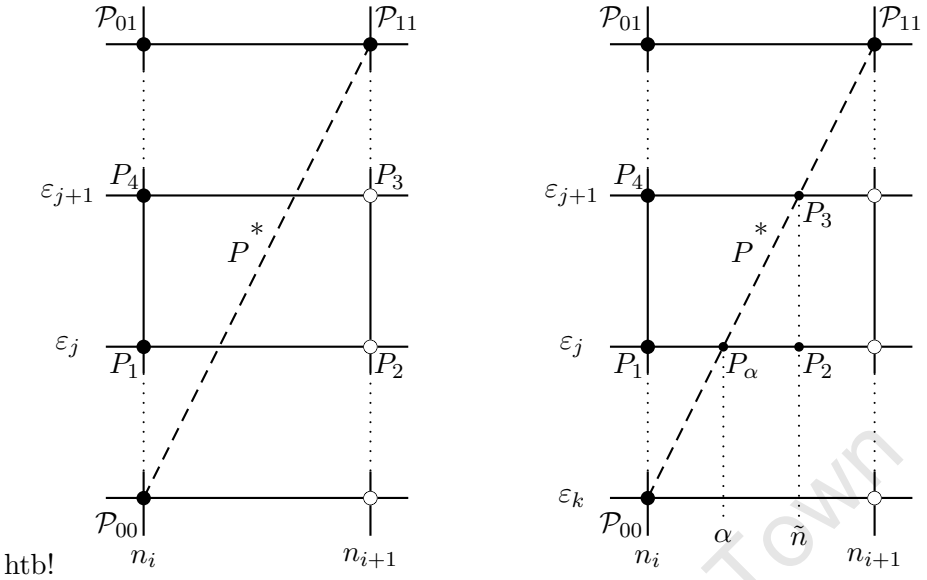


Figure 3.1 – Contour plot for pressure, as a function of energy and baryon density, for a hadron gas. The white region is the region of non-existence.

you might have the situation where the P_2 and possibly P_3 do not exist. We illustrate this with the diagram on the left in Fig 3.2. The closed circles are points on the grid



htb!

Figure 3.2 – Illustration of existence edge (dashed line) in (n, ε) plane. The open circles are points where the pressure does not exist, while at closed circles the pressure does exist

where the pressure exists while the open circles are outside the region of existence, i.e. the pressure does not exist. In this situation we "restructure" the grid around the point of interest and estimate ghost values for P_3 and P_2, as shown in diagram on the right in Fig 3.2. To estimate the value of P_3, we assume that the pressure values along the boundary line (dashed line) have the same behaviour as those above $n = n_i$, i.e.

$$\frac{P_3 - \mathcal{P}_{00}}{\mathcal{P}_{11} - \mathcal{P}_{00}} = \frac{P_4 - \mathcal{P}_{00}}{\mathcal{P}_{01} - \mathcal{P}_{00}}$$

which gives

$$P_3 = \mathcal{P}_{00} + \left(\frac{\mathcal{P}_{11} - \mathcal{P}_{00}}{\mathcal{P}_{01} - \mathcal{P}_{00}}\right)(P_4 - \mathcal{P}_{00})\,. \tag{3.15}$$

The initial assumption was to then assume that the point P_2, lay on the same plane as points P_1, P_3 and P_4, this however gave unsatisfactory results as discontinuous behaviours are obtained between adjacent grid cells. Lets assume that the point of interest lies on the boundary line (dashed line). In this situation there will be a discontinuous jump in the value for P as the point crosses the boundary between two cells of the grid.

A better choice is to fix P_2, such that the bilinear fit reproduces the same value for the point (α, ε_j), i.e. P_α, as equation (3.15). This then yields

$$P_2 = \frac{(\varepsilon_{j+1} - \varepsilon_k)P_\alpha - (\varepsilon_{j+1} - \varepsilon_j)P_1}{(\varepsilon_j - \varepsilon_k)} \tag{3.16}$$

where from equation (3.15)

$$P_\alpha = \mathcal{P}_{00} + \left(\frac{\mathcal{P}_{11} - \mathcal{P}_{00}}{\mathcal{P}_{01} - \mathcal{P}_{00}}\right)(P_1 - \mathcal{P}_{00})\,.$$

now the values for P along the boundary line are continuous across the grid cell boundaries.

TRelHydro makes use of the tabulated equation of state along with a set of fitting functions to determine the values of P and v for a given set of values for T^{00}, T^{0x} and $\tilde{N}$.

3.1.2 Freeze-Out

In (1+1) dimensions the freeze-out hypersurface becomes a freeze-out curve, and as such can be parametrised by one parameter, $\sigma^\mu = [t(\xi), x(\xi)]$. The outward normal to the surface now becomes

$$\begin{aligned} \mathrm{d}\sigma^\mu &= -\epsilon_{\mu\nu}\frac{\partial\sigma^\nu}{\partial u}du \\ &= -\left[\frac{\partial x}{\partial\xi}, \frac{\partial t}{\partial\xi}\right]\mathrm{d}\xi, \end{aligned} \tag{3.17}$$

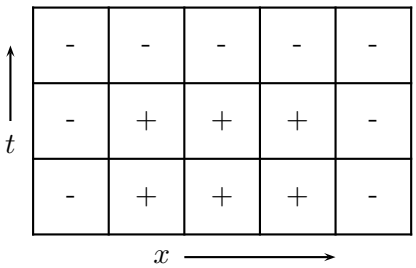
where epsilon is the Levi-Civita symbol and the curve has positive orientation for increasing ξ. Thus, if we have the freeze-out curve, the number of particles which have frozen out is given by

$$N = -\int_\sigma \gamma n\left(\frac{\mathrm{d}x}{\mathrm{d}t} - 1\right)dt \tag{3.18}$$

where $\dfrac{\mathrm{d}x}{\mathrm{d}t}$ is the slope of the curve and the curve is orientated positively.

In order to perform this freeze-out we require the freeze-out curve, which is determined by the freeze-out criteria. In this hydrodynamic application, freeze-out is chosen to occur when the temperature reaches a freeze-out temperature T_{fo}. At each timestep we step through the grid looking for adjacent cells whose temperatures are on either side of T_{fo}. A linear interpolation is done between the temperatures of the two cells to determine the spatial coordinate corresponding to temperature T_{fo}. Linear interpolations are then also used to calculate the energy density, baryon density, particle density and flow velocity at this point. We also set a flag to indicate the direction of the normal.

There may be regions where "numerical" instantaneous freeze-out occurs. This is scenario is shown in the diagram below.



The "+" and "−" signs indicate that the temperature is either above or below T_{fo}. In this situation, there is instantaneous freeze-out for the middle cell in the second row. In this case, we interpolate linearly between this cell and the one above it to find the *temporal* coordinate which corresponds to the value T_{fo}. Again we interpolate to find the values of flow velocity, energy, baryon and particle density at this point.

By performing the linear interpolation, we obtain a much smother curve than taking the first order choice of choosing the interface between the inside and outside cell as the point on the freeze-out curve.

At the end of the simulation the rapidity spectra of the particles can be created using the stored data on the freeze-out curve (§3.3.2). There is the option to perform the freeze-out online (while the simulation is running), but at this stage the matter frozen out is cold, it has no thermal distribution.

3.1.3 Initial Condition

TRelHydro has a built in initial condition for setting the initial profiles of the distributions. This initial condition consists of static Wood-Saxons distributions for the energy density

$$\varepsilon = \frac{\varepsilon_0}{1 + e^{\frac{|x|-r_0}{\beta}}}, \tag{3.19}$$

and baryon density

$$n = \frac{n_0}{1 + e^{\frac{|x|-r_0}{\beta}}}, \tag{3.20}$$

with half-width, $r0$, skin-width, β, and maxima ε_0 and n_0, respectively. All the other distributions are then calculated from these and the fact that the velocity $v = 0$. It is, however, possible to use load ones own initial profiles for the energy density, baryon density and velocity.

3.2 The Code

TRelHydro consists of the main class, **TRelHydro**, a container class for parameters, **TRHParam**, a class for performing freeze-out operations, **TRHFO**, and a base class for

the equation of state, **TRHEOS**. Having this base class for the equation of state allows one to code various interchangeable equation of states and introduces the functionality of being able to load any equation of state you wish to use at run time. Thus one can code their own equation of state class as long as it contains implementations of the virtual functions defined in **TRHEOS** class. The various interchangeable equation of states which are currently included are implemented through the classes; **TRHBagEOS**, **TRHMassless**, **TRHThmCFO** and **TRHTabEOS**.

The descriptions of the classes follow the same format as for those of **FCTHydro**: For each class we review the important members and member functions. The members are listed with the name of the member followed by the data type of the member and then a description of it. The functions are listed with the name of the function and its input variables and followed by its output type. The use of each function is also provided. At the end of each section there is also a diagram indicating the order in which the functions must be called. An arrow means that the function(s) to the left of the arrow must be called before the function(s) to the right of the arrow. Any input variable with an "s" prefix indicates that it is used for setting the value of a member of the class.

3.2.1 TRHParam

This is a container class for the various parameters which are used in the system and simplifies the setting and checking of parameters at run time. It's members are

N	*Int_t	number of physical cells
dx	*Double_t	initial size of each cell
dt	*Double_t	timestep
hdt	*Double_t	half of timestep
$x0$	*Double_t	lower edge of grid
xn	*Double_t	upper edge of grid
$writeInterval$	*Int_t	number of iterations to run between each output of data
$noWrite$	*Int_t	number of times to output data
$time$	*Double_t	time at end of time step
$twoStep$	*Int_t	bool switch for the twostep method
$e0$	*Double_t	maximum value of initial energy density
$b0$	*Double_t	maximum value of initial number density
a	*Double_t	variable for setting skin-width of Wood-Saxons distribution
$r0$	*Double_t	variable for setting half width of Wood-Saxons distribution
$toll$	*Double_t	numerical tolerance for vacuum
$outFileDir$	*TString	directory to which data is written

You will notice that all of the members are in fact pointers to the values of the parameters. The total number of timestep iterations is determined by the product of $writeInterval$ and $noWrite$. The parameters $e0$, $b0$, $r0$ and a are used in the built in initialisation script of **TRelHydro**. This script initialises the energy density and baryon density to a Wood-Saxons distribution with halfwidth $r0$, skinwidth $a \times dt$ and maximum values $e0$ and $b0$, respectively. The most useful function is

SetParams(ifstream *$confile$) void

which takes a pointer to a configuration file containing the values for the parameters. See section §3.2.5 for information on the layout of this file. There are also "Get" functions for all of the parameters which return pointers their values.

3.2.2 TRHEOS

This is the base class from which all the various interchangeable equation of state classes inherit. It contains the virtual function definitions of the functions required by the classes which use the equation of state. It also has the common parameters for all the equation of state classes;

N	Int_t	size of the system
$Toll$	Double_t	numerical tolerance for vacuum
$Temp$	**FCTVector**	vector for temperature profile

The virtual functions required are

SetParams(ifstream *$confile$)	virtual TString
SetVectorSize(Int_t sN)	virtual void
CalcVelEpsPresTemp(Double_t *Nbnr, Double_t *T00, Double_t *T0x,Double_t *vel, Double_t *eps, Double_t *bnr, Double_t *pres)=0	virtual Int_t
InitPresTemp(Double_t *eps,Double_t *bnr,	virtual void
GetEntropy(Int_t i, Double_t *eps, Double_t *bnr, Double_t *pres)=0	virtual Double_t
GetNumDens(Int_t i, Double_t *eps, Double_t *bnr, Double_t *pres)=0	virtual Double_t
SetToll(Double_t sToll);	virtual void

The function SetParams() takes a pointer to a configuration file to set $Toll$, else SetToll() must be called. As usual SetVectorSize() is used to set the size of the system. The function InitPresTemp() is used to calculate the initial pressure and temperature profiles of the system given the known initial local energy density and local baryon density profiles. GetNumDens() returns the number density given that the local energy and baron densities are known. During the evolution of a relativistic hydrodynamical system the equations of motion are used to evolve the components of the energy-momentum tensor and the lab-frame baryon density. CalcVelEpsPresTemp() calculates the flow velocity, local energy and baryon density the pressure and the temperature, given values for the lab-frame baryon density and energy-momentum tensor components. There are currently four different types of equations of state which have been coded. As the classes below all inherit from this **TRHEOS** class, they all contain the above functions and members, however, their implementations vary.

What follows is the descriptions of the four equation of states which are currently included in **TRelHydro**. For each one the additional parameters and functions which they require, above those already mentioned, are described.

TRHMassless

This contains the routines for a thermal gas of massless Boltzmann, Fermi-Dirac or Bose-Einstein particles, or mixtures there of. The extra members which are needed by this class are

$Qstats$	Double_t [3]	array of fractions of gas which are [Boltzmann, Fermi-Dirac, Bose-Einstein]
g	Double_t	degeneracy factor

And the extra functions are

SetQStatsFrac(Double_t Boltz, Double_t FD, Double_t BE)	void
PrintQstats()	void
SetG(Double_t sg)	void

SetParams() now also sets g, else SetG() must be called. SetQStatsFrac() is used to set the fraction of the gas which is Boltzmann, Fermi-Dirac and Bose-Einstein. It performs a check to verify whether the sum of the three values adds to one. PrintQstats() outputs

the fractions to standard output and is needed as $Qstats$ is a private member.

The ordering of the function calls:

$$\text{SetParams()or}\left\{\begin{array}{l}\text{SetToll()}\\ \text{SetQStatsFrac()}\\ \text{SetG()}\\ \text{SetVectorSize()}\end{array}\right\}\rightarrow \text{InitPresTemp()}\rightarrow\left\{\begin{array}{l}\text{CalcVelEpsPresTemp()}\\ \text{GetEntropy()}\\ \text{GetNumDens()}\end{array}\right.$$

TRHThmCFO

This class performs the calculations for a Grand Canonical thermal hadronic gas equation of state by interfacing with the THERMUS package [101, 105]. This thermal equation of state can have a baryon (μ_B) and strangeness (μ_S) chemical potential. At each point on the grid for each time interval, this class tries to find the temperature, flow velocity and baryon and strangeness chemical potentials for given values of the lab-frame baryon density and components of the energy-momentum tensor. There are a collection of external fitting functions which make use of the "broyden" and "newt" routines from the Numerical Recipes libraries [106]. Broydens Method is a generalisation of the secant method to higher dimensional nonlinear systems and "newt" uses a globally convergent multi-dimensional Newton's method. The external fitting functions are located in the file "HOESfuncts.cxx".

Some important members of this equation of state class are

$fitVel$	Double_t	value for the flow velocity fitted by the externalfunctions
$Widths$	Bool_t	switch for using widths in the thermal model
$Qstats$	Bool_t	switch for using quantum stats in the thermal model
$ParticleList$	TString	file containing the particles to be loaded in the thermal model
$bFitEpsCut$	Double_t	local energy density value above which the fitting is done
$broydnStartTemp$	Double_t	starting temperature value for the broyden fit
$broydnStartMuB$	Double_t	starting μ_B value for the broyden fit

broydnStartMuS	Double_t	starting μ_S value for the broyden fit
MuB	**TRHVector**	vector for μ_B
MuS	**TRHVector**	vector for μ_S
constrainMus	Bool_t	switch for constraining μ_S with strangeness density set to zero
pset	**TTMParticleSet**	Thermal Particle Set object
thParams	**TTMParameterSetBSQ**	Thermal Parameter Set object
thm	**TTMThermalModelBSQ**	Thermal Model object

TTMParticleSet, **TTMParameterSetBSQ** and **TTMThermalModelBSQ** are THERMUS classes. They are the classes related to the Grand Canonical Ensemble which consists of the particles listed in the **TTMParticleSet** object. The following are important member functions of **TRHThmCFO**;

SetParticleSet(TString *ParticleList*);	void
SetTMuBMuS(Double_t sT=0.1, Double_t *smub*=0.0, Double_t *smus*=0.0)	void
SetWidths(Bool_t *withWidths*)	void
SetQStats(Bool_t *sQstats*)	void
SetbFitEpsCut(Double_t *sbFitEpsCut*)	void
SetBroydnStartValues(Double_t sT, Double_t sMb, Double_t sMs=0.0)	void

SetParams() is used to set the values for *Widths*, *Qstats*, *ParticleList* and *bFitEpsCut*, else SetWidths(), SetQStats(), SetParticleSet() and/or SetbFitEpsCut() must be called. As usual SetVectorSize() is used for setting the size of the **FCTVector** objects in the class. SetTMuBMuS() sets the values of the temperature, μ_B and μ_S in the thermals model parameter object, *thParams*. The function SetBroydnStartValues() sets the starting values of the temperature, μ_B and μ_S or the broyden routine.

$$\text{SetParams()or}\left\{\begin{array}{r}\left\{\begin{array}{l}\text{SetToll()}\\ \text{SetParticleSet()}\\ \text{SetWidths()}\\ \text{SetQStats()}\\ \text{SetbFitEpsCut()}\end{array}\right.\\ \text{SetVectorSize()}\\ \text{SetTMuBMuS()}\\ \text{SetBroydnStartValues()}\end{array}\right\}\rightarrow \text{InitPresTemp()}\rightarrow\left\{\begin{array}{l}\text{CalcVelEpsPresTemp()}\\ \text{GetEntropy()}\\ \text{GetNumDens()}\end{array}\right.$$

The down side to this equation of state class, is that at each time step and for each point on the grid, a fit is performed for the temperature and other thermodynamic parameters. Since for each step of the fitting process THERMUS does a number of multidimensional integrations, the numerics can be very computational intensive. The speed of the simulation can be dramatically improved with the use of a tabulated equation of state. Although this is less accurate, the dramatic increase in speed is well worth it. Thus if you would like to do simulations with a the thermal equation of state above, it is preferable to do one set of fits to create the table, then use the table for various initial conditions.

TRHqgEOS

This class performs the necessary calculations for the equation of state of the gass which consists of quarks and gluons at higher temperatures and massless pions at lower temperatures. There is also a mixed phase during the first order phase change between these two states. It has the extra members

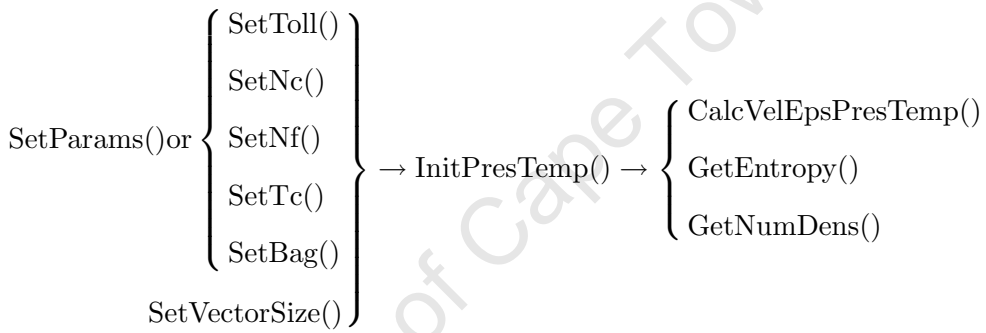
Nc	Double_t	the number of colours
Nf	Double_t	the number of flavours
Bag	Double_t	the bag constant
Tc	Double_t	the critical temperature for the phase change

And extra functions:

SetNc(Double_t sNc)	void
SetNf(Double_t sNf)	void

SetBag(Double_t $sBag$)	void
SetTc(Double_t sTc)	void

The function SetParams() also sets Nf and Nc, and either Tc or Bag from the configuration file values. If these are not specified in the file then SetNf(), SetNc() and/or SetBag() or SetTc() must be called. Since the values of Tc and Bag are related; if you set the critical temperature Tc using SetTc() it also calculates the corresponding value for Bag. Similary if Bag is set using SetBag() then the corresponding value for Tc is calcualted. Thus one either runs SetTc() or SetBag(), and not both. If both a value for Bag and Tc are specified in the configuration file, then Tc takes precidence.



TRHTabEOS

This class was written for simulating a system using a tabulated equation of state. This class uses the fitting routines "newt" and "zbrent" of the Numerical Recipes in C libraries [106]. The routine "zbrent" finds the root of a function using Brent's method. Again there are external fitting functions which use these routines and they are collated in the file "TABfuncts.cc". The tabulated equation of state provides the temperature, number density, pressure, μ_B, μ_S and entropy density as a function of energy density (ε) and baryon density (n_B).

Instead of the keeping the values in two-dimensional histograms we opted for creating an array of **BDensObj** objects. Each of these objects correspond to a particular value of the baryon density and contains an array for energy density values and arrays for the corresponding values of pressure, temperature, μ_B, μ_S and entropy density. It is important that the values of the corresponding *Edens* (energy density) elements of different **BDensObj** objects have the same value*. See below for a description of the

*See Fig C.1 for an illustration.

BDensObj class. The members which are needed by **TRHTabEOS** are

MuB	**FCTVector**	vector for μ_B profile
MuS	**FCTVector**	vector for μ_S profile
$Bdens$	**TObjArray**	array for **BDensObj** objects
$BLastPos$	Int_t	position in the object array just below fitted n_b
$ELastPos$	Int_t	position in the energy density array just below fitted ε

And the extra functions needed are:

WriteToFile(TString $TabFName$)	void
LocateBnrEpsPosition(Double_t e,Double_t nb)	Int_t
GetNbins()	Int_t
GetBLastPos()	Int_t
GetELastPos()	Int_t
GetMuB(Int_t i)	Double_t
GetMuS(Int_t i)	Double_t
GetBDens(Int_t i)	***BDensObj**

SetVectorSize() now also sets the size of MuB and MuS. As the name suggests, WriteToFile() outputs the tabulated eos to the file with name $TabFName$. GetNbins() returns the number of **BDensObj** objects in the array $Bdens$. GetBLastPos() and GetELastPos() return the values of $BLastPos$ and $ELastPos$, respectively, as these are private members. GetMuB() and GetMuS() return the i'th component of MuB and MuS, respectively. LocateBnrEpsPosition() sets the appropriate values of $BLastPos$ and $ELastPos$ given energy and baryon density values, ε and nb. GetBDens() returns a pointer to the i'th **BDensObj** object in the $Bdens$ object array.

SetParams() or SetToll()
SetVectorSize() → InitPresTemp() →
- CalcVelEpsPresTemp()
- GetEntropy()
- GetNumDens()
- GetNbins()
- GetMuB()
- GetMuS()
- LocateBnrEpsPosition()
- WriteToFile()

You will notice that there is no function for reading a tabulated equation of state from a file. This is because everyone uses their own personal format when writing tabulated data to file. One must create an instance of this class and then fill the **TObjArray** in the appropriate manner. This can be achieved with a ROOT macro[†] to load your specific tabulated eos into the **TObjArray**. This instance of **TRHTabEOS** can then be saved in a ROOT file and thus loaded when it is needed by a hydrodynamic simulation.

BDensObj

This is the container class for the tabulated equation of state class **TRHTabEOS**. The **TObjArray** *Bdens* contains instances of this class. The important members are

Ebins	unsigned int	number of components in the arrays below
density	Double_t	baryon density value
Edens	*Double_t	pointer to energy density array
T	*Double_t	pointer to temperature array
Mub	*Double_t	pointer to μ_B array
Mus	*Double_t	pointer to μ_S array
Pres	*Double_t	pointer to pressure array
Ent	*Double_t	pointer to entropy density array
Num	*Double_t	pointer to number density array

We opted to use arrays as opposed to instances of **FCTVector** as each of the vector parameters has the same number of elements. The functions which this **BDensObj** class use are:

BDensObj(unsigned int *sN*)	
SetSize(unsigned int *sN*)	void
WriteToFile(ofstream **ofile*, Bool_t *printheaders*)	void

SetSize() sets the size of the arrays and the value of *Ebins* to *sN*. The constructor BDensObj(unsigned int *sN*) creates an instance of **BDensObj** and then runs SetSize() with the argument *sN*. The function WriteToFile() outputs the data of the instance to the output file with the pointer **ofile*. The bool variable *printheaders* informs the function to precede the data in the file with a header explaining the output format, if set to **true**.

3.2.3 TRHFO

This class contains all of the necessary functions and members needed for calculating the freeze-out hypersurface and storing related data. It makes use of the container class **TRHFOData**. **TRHFO** calculates the position of points along the freeze-out curve as

[†]See Appendix C for an example of this

well as the flow velocity, number density and information regarding the direction of the outward normal to the freeze-out curve at these points. The main parameters for this class are

eos	***TRHEOS**	pointer to the object for the equation of state
N	Int_t	size of the system
dt	Double_t	timestep
Tfo	Double_t	freeze-out temperature
$fosurfFile$	ofstream	output file for the data points of the freeze-out curve
$dndyFile$	ofstream	output file for the $\frac{\mathrm{d}N}{\mathrm{d}y}$ data
$oldVel$	**FCTVector**	velocity profile for previous time step
Num	**FCTVector**	particle number density profile
$oldNum$	**FCTVector**	number density profile for previous time step
$oldTemp$	**FCTVector**	temperature profile for previous time step
$dNdy$	**TH1D**	histogram of $\frac{\mathrm{d}N}{\mathrm{d}y}$
$foData$	**TRHFOData** [2]	the containers for the freeze-out points

The first component of $foData[0]$ contains the data points of the freeze-out curve from the previous timestep while the second component $foData[1]$ contains the points for the current timestep. The important member functions are

SetParams(ifstream *$confile$)	void
SetVectorSize(Int_t sN)	void
SetTfo(Double_t $sTfo$)	void
SetDeltat(Double_t sdt)	void
SetEOS(TRHEOS *eos)	void
FindFreezeOutCurve(Double_t *x, Double_t *eps, Double_t *vel, Double_t *$pres$, Double_t *bnr, Double_t $time$)	void
InitFOData(Double_t *x, Double_t *eps, Double_t *vel, Double_t *$pres$, Double_t *bnr, Double_t $time$)	void
FreezeOutParticles()	void
StoreOldValues(Double_t *vel)	void
OutPutData()	void
OpenDataFile(TString $DataFilesDir$)	void

SetParams() sets the value of Tfo using the file with pointer $confile$ else SetTfo() must be called. As usual, SetVectorSize() sets the size of **FCTVector** objects. SetEOS() sets the pointer eos to point at the equation of state instance which is being used. This is needed to access to the temperature profile and calculate the particle number density. The function InitFOData() finds the points of the freeze-out surface at the start of the simulation while FindFreezeOutCurve() calculates the points on the freeze-out curve at the current moment. The function FreezeOutParticles() freezes out the particles

through the freeze-out hypersurface to calculate the $\frac{\mathrm{d}N}{\mathrm{d}y}$ spectra. OpenDataFile() opens the output files with names "dNdy.dat" and "fosurf.dat" in the directory specified by $DataFilesDir$ and OutPutData() writes the data to disk.

$$\left.\begin{array}{r}\text{SetParams() or SetTfo()}\\ \text{SetVectorSize()}\\ \text{SetEOS()}\\ \text{SetDeltat()}\\ \text{OpenDataFile()}\end{array}\right\} \rightarrow \text{InitFOData()} \rightarrow \text{FindFreezeOutCurve()} \cdots$$

$$\cdots \rightarrow \text{FreezeOutParticles()} \rightarrow \text{OutPutData()}$$

TRHFOData

This is a container class used by the **TRHFO** class. It contains the coordinates which define the freeze-out curve as well as the value of the flow velocity and local number density at these coordinates. There is also a variable which describes the direction of the outward normal of the freeze-out curve. The important members of this class are

N	Int_t	number points defining the freeze-out curve
$Xpos$	**FCTVelocity**	vector of spatial coordinates of the points on freeze-out curve
$Tpos$	**FCTVelocity**	vector of temporal coordinates of the points on freeze-out curve
Vel	**FCTVelocity**	vector of flow velocity values at the points on freeze-out curve
Num	**FCTVelocity**	vector of flow velocity values at the points on freeze-out curve
$Norm$	**FCTVelocity**	vector of values defining the direction of the outward normal to the freeze-out curve
Loc	**FCTVelocity**	vector of the location in the grid of the cells whose centres are just outside the freeze-out curve

The functions which are needed when using this class are:

ClearData()	void
operator=(const **TRHFOData**& $sfodata$)	**TRHFOData**&
AddFOpoint(Double_t sxp, Double_t $snum$, Double_t $svel$,	void

Double_t *snorm*, Double_t *sloc*, Double_t *stp*)

GetN() Int_t

ClearData() empties the vectors and sets N to 0. The function GetN() simply returns the value of N and the operator "=" sets the current instance to have the exact same values as the **TRHFOData** object $sfodata$. Finally the most useful function is AddFOpoint() as this function adds the data for a new point on the freeze-out curve found by the functions of **TRHFO**.

3.2.4 TRelHydro

Now we can now describe the main class **TRelHydro** and how it simulates a (1+1) dimensional, relativistic hydrodynamic system with the aid of the classes **TRHParam**, **TRHEOS**, **TRHFO**. As we saw at the beginning of this chapter, it is the lab-frame baryon density $\tilde{N}_B$ and the energy-momentum tensor components T^{00} and T^{0x}, which are evolved.
The important members of **TRelHydro** are:

N	Int_t	number of physical cells
EOS	***TRHEOS**	pointer to eos object
FO	**TRHFO**	performs freeze-out functions
$params$	**TRHParam**	contains parameter values
$velocity$	**TRHVelocity**	calculates velocity dependent parameters
$grid$	**TRHGrid**	calculates geometry parameters
$T00$	**FCTFluid**	for evolving T^{00}
$T0x$	**FCTFluid**	for evolving T^{0x}
$Nbnr$	**FCTFluid**	for evolving lab-frame baryon density
$Fluids$	**THashList**	list containing **FCTFluid objects to simplify coding**
eps	**FCTVector**	local energy density profile
bnr	**FCTVector**	local baryon density profile
$t00SourceD$	**FCTVector**	profile of the source for T^{00} evolution
$t0xSourceD$	**FCTVector**	profile of the source for T^{0x} evolution
$entropy\_diff$	Bool_t	switch for calculating entropy production
$OFNames$	*TString	array of output file names
$DataOutType$	TString	specifies the type of data output
$DataLocation$	TString	directory in which to output data
$heps$	***TH2D**	stores local energy density profiles
$hpres$	***TH2D**	stores pressure profiles
$hvel$	***TH2D**	stores velocity profiles
$hbnr$	***TH2D**	stores local baryon density profiles
$ht00$	***TH2D**	stores T^{00} profiles
$ht0x$	***TH2D**	stores T^{0x} profiles

htemp	***TH2D**	stores temperature profiles
hmub	***TH2D**	stores μ_B profiles
hmus	***TH2D**	stores μ_S profiles
hent	***TH2D**	stores entropy profiles
hdNdy	***TH2D**	stores $\frac{\mathrm{d}N}{\mathrm{d}y}$ profiles
vel	*Double_t	pointer to velocity profile
$t00$	*Double_t	pointer to T_{00} profile
$t0x$	*Double_t	pointer to T_{0x} profile
x	*Double_t	pointer to position of cell centres

For each density which is evolved, an associated **FCTFluid** object is created. Then **THashList** *Fluids* is used when the same call must be made to all the **FCTFluid** objects. With this formulation, the addition of extra conserved currents to the system is fairly simple. There are two options for the data output; file (*DataOutType*=File) and histogram (*DataOutType*=Hist). If the histogram type is chosen then the **TH2D** objects are used for capturing the various profiles else they are written to file. This is done every *writeInterval* (section §3.2.1) time steps. The pointers *vel*, $t00$, $t0x$ and x are introduced to increase the readability of the code.

The member functions which must be called to simulate the system are:

TRelHydro(**TRHEOS** *$sEOS$,TString $confFile$)	
SetEOS(**TRHEOS** *$sEOS$)	void
SetParams(TString $confFile$)	Int_t
Initialise()	void
Initialise(**TRHVector** *$sEps$, **TRHVector** *$sVel$, **TRHVector** *$sBnr$)	void
ConvectSystem(Bool_t *verbose*, Bool_t $freezeout$ = **false**)	Int_t
SetOutPutTypeFile()	void
SetOutPutTypeFile(TString $DataFilesDir$)	void
SetOutPutTypeHist()	void
OutputResults(Int_t k)	void
SetEntropy_diff(Bool_t $sentropy\_diff$, TString $sEntFile$)	void
CalcEntropyDiff()	void
WriteHist(TString $DataFilesDir$)	void
WriteHist()	void

The function SetEOS() sets the equation of state you wish to employ by pointing EOS at the instance of an equation of state class. SetParams() sets the parameters of the system using the values specified in the configuration file with name $confFile$. See section §3.2.5 for the required format of the input data in the configuration file. TRelHydro() is the constructor which then calls SetEOS() and SetParams() with the arguments $sEOS$ and $confFile$, respectively. The function SetEOS() must be called before SetParams(), as SetParams() passes a pointer to the configuration file to the other classes used by **TRelHydro**

Initialise() with out any arguments initialises the system such that the local rest-frame energy and baryon densities have static Wood-Saxon distributions with half-width $r0$, skin width $a \times dx$ and maximum values $e0$ and $b0$, respectively (see section §3.1.3). This is done on a uniform grid with lower boundary at $x0$ and grid spacing dx. These profiles are then used to calculate the lab-frame baryon density, T_{00}, T_{0x}, temperature profiles, as well as any other profiles required by the equation of state class. The values of these parameters are stored in the **TRHParam** instance, *params*.
If the function is called with the arguments ($sEps$, $sVel$, $sBnr$) then it initialises the system with a regular grid with lower boundary $x0$, grid spacing dx and the profiles of the flow velocity and the local rest-frame energy density and baryon density as specified by the vectors with pointers $sVel$, $sEps$ and $sBnr$, respectively. The other profiles required are then calculated from these.

The functions SetOutPutTypeFile() and SetOutPutTypeHist() set the output type to either an ASCII file or ROOT histograms, respectively. The ASCII output files are located in the directory specified by the parameter $outFileDir$ (section §3.2.1). The output directory can be over ridden by calling SetOutPutTypeFile() with the argument $DataFilesDir$, a string with the path of the directory for the data output. The code outputs the data at a certain timestep, according to the chosen style, by calling the function OutPutData() with the argument k denoting which output it is in the sequence of outputs. During the evolution, snapshots of the profiles are taken after a certain number of specified iterations and stored in two dimensional histograms.The function WriteHist() is called at the end of the evolution to write these histograms to a ROOT file, in the directory specified by $outFileDir$ or $DataFilesDir$.

SetEntropy_diff() is used to toggle the codes calculation of entropy production. The bool variable $sentropy\_diff$ is the switch and $sEntFile$ is the name of the file to which the output must be written. If $sentropy\_diff$ is **true** then CalcEntropyDiff() determines the increase in entropy between the last timestep and this timestep. This is very useful as a consistency check when simulating an ideal hydrodynamical system as there should be no entropy production. Any increase in the entropy is then due to the numerical errors alone.

Finally the function ConvectSystem() is called to simulate the evolution of the system. The bool argument *verbose*, is a switch for outputting information to the standard output every timestep as opposed to every $writeInterval$ timesteps. The argument $freezeout$ is a bool switch for calculating the freeze-out curve and performing the freeze-out. This is the function with the code structure for implementing the twostep method if $twoStep = 1$.

$$\left.\begin{array}{l}\text{SetEOS()} \rightarrow \text{SetParams()} \\ \text{SetOutPutTypeFile()} \\ \quad or \\ \text{SetOutPutTypeHist()}\end{array}\right\} \rightarrow \left.\begin{array}{l}\text{Initialise()} \\ \text{SetEntropy\_diff()}\end{array}\right\} \rightarrow \left\{\begin{array}{l}\text{ConvectSystem()} \\ \text{CalcEntropyDiff()} \\ \text{OutputResults()} \\ \text{WriteHist()}\end{array}\right.$$

3.2.5 How To use

To gain a better understanding of how to simulate a system using **TRelHydro** see the ROOT macros below

```
gSystem->Load("../../FCTHydrolib/libFCTHydro.so");
gSystem->Load("../lib/libRHMassless2.so");
gSystem->Load("../lib/libRelHydro2.so");

TRHMassless EOS;
EOS.SetQStatsFraction(0.0,0.0,1.0);

TRelHydro bob(&EOS,"Config.file");

bob.grid.InitialseGrid();
gROOT->ProcessLine(".x init_hydro4.c");
bob.Initialise(&ee,&vv,&bb);

bob.SetEntropy_diff(true,"./datafile/entropy.dat");
bob.SetOutPutTypeFile();
bob.OutputResults(0);

bob.ConvectSystem(false,true);
```

Firstly, load the the libraries for **FCTHydro** and the equation of state you wish to use, in this case a gas of massless particles. Then load the library for the main class **TRelHydro** and its necessary components. Once the various libraries are loaded instantiate the equation of state class. Since the massless particle equation of state is being use, the fractions of the gas which obey Boltzmann, Fermi-Dirac and Bose-Einstein statistics must be set. In this case the gas is purely Bose-Einstein. Now that the equation of state class has been initialised the **TRelHydro** class, here *bob*, can be initialised. The file with name "Config.file" is the configuration file with the values of the parameters (section §3.2.5).

The next three lines of code are for initialising the profiles. If all we had wanted were static Wood-Saxon distributions for the local rest frame energy and baryon densities then instead of these three lines one would just call Initialise(), with no argument. For a better understanding of the scope of this code, we present an example where the initial profiles are determined outside of **TRelHydro**. You might notice that bob.grid.InitialseGrid() is called, this initialises the grid for use in another macro which determines the initial profiles. The next line runs the macro "init_hydro4.c" (shown below) which sets up a static square wave for the local rest-frame energy density and sets the baryon density to zero. These profiles are then passed to *bob* and the system is initialised.

The next line of code tells *bob* to calculate the entropy production and output the data to the file "./datafiles/entropy.dat". Then *bob* is told to output the profiles to an ASCII file in the directory specified in the container *params*. This is followed by the call to OutPutData() with the argument "0", which tells *bob* to output the the initial profiles as the 0^{th} output. Finally, *bob* is told to evolve the system without outputting verbose data and to perform the freeze-out calculations.

Below is the macro "init_hydro4.c" for calculating the initial profiles:

```
int nn=bob.N+2;
double hdx =*bob.params.GetDx()/2.0;
double *xx = bob.x;
TRHVector ee(nn),vv(nn),bb(nn);
double e0=0e-16;
double e1=43.0308388;
for (int i=0;i<nn;i++)
{
  if (TMath::Abs(xx[i]) < 1.0+hdx) ee[i]=e1;
  else ee[i]=e0;
}
```

Note that when a **TRHVector** class is instantiated without a size, then all the elements of the vector are automatically zero.

Configuration file

Most of the classes listed above can get their parameter values from a configuration file. This input file must have the data in a certain format. The name of the parameter appears first and is followed by a equal sign "=", then by the parameters value. These must all be separated by spaces. Anything that occurs after the hash symbol "#" is ignored. For example,

```
residualDiff = 0.9       # this text is ignored
```

sets the value of the parameter $residualDiff$ to 0.9. See appendix B for a configuration file which can be used for this (1+1)D system. The system only looks for parameters that are needed and ignores the others. For example if we are using the equation of state $TRHMassless$ and the and there are values set for parameters of $TRHThmCFO$, then the latter's parameters are ignored.

3.3 Applications and Testing

In this section we present the solutions obtained from **TRelHydro** for systems with known analytic results. These provide indications of the accuracy of **TRelHydro** and thus for relativistic hydrodynamic simulations using **FCTHydro**.

3.3.1 Shock problem

The first test is a relativistic shock tube problem described in [107]. This is an extreme system as the shock waves introduce large gradients and provides a tough constraint on the numerics. If **FCTHydro** performs well for these initial conditions it should preform well for most systems setups. The solutions obtained from **TRelHydro**, using different values for the parameters, are compared to the analytic solution. The initial conditions consist of two touching static regions, Fig 3.3(a) with the first region at pressure $P_0 = 1.0$

and baryon density $n_0 = 1.0$ and the second at pressure $P_4 = 0.1$ and baryon density $n_4 = 0.125$. For the simulation a Boltzmann massless gas was used, i.e. the equation of state is $P = \frac{\varepsilon}{3}$.

When the evolution begins, a rarefaction wave travels into the region of higher pressure while a shock wave travels into the region of lower pressure. There are five separate regions which can be identified as seen in Fig 3.3(b). The left most region is where the

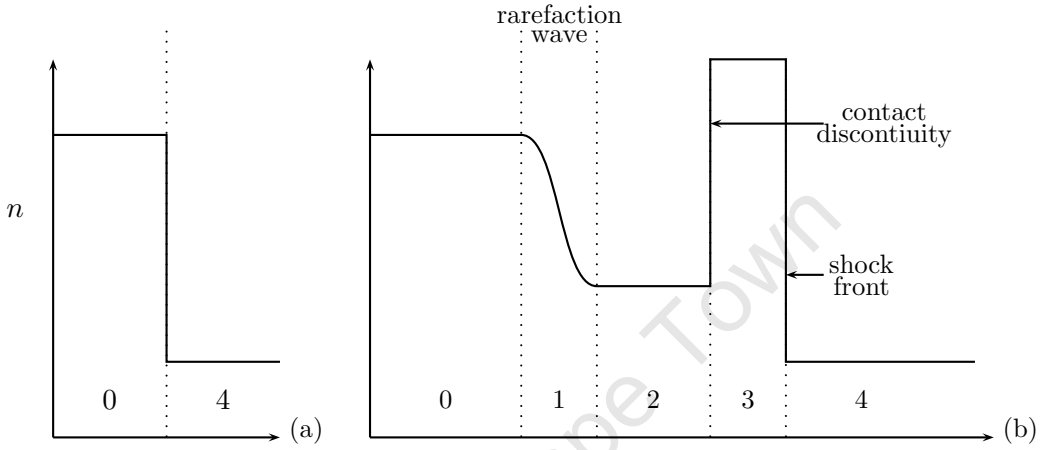


Figure 3.3 – An illustration of the shock problem: The initial conditions (a) and the solution after time t (b).

fluid at the higher pressure is still undisturbed. Then we have the rarefaction wave which is travelling left. This is followed by a constant state between the rarefaction wave and the contact discontinuity which trails the shock front travelling to the right. Finally we have the fluid at the lower pressure which is still undisturbed. The details of the analytic solution can be found in [107].

A grid with $N = 1000$ points and grid spacing $dx = 0.025$ fm was used for all the tests and the results are presented at time $t = 3.75$ fm/c. With this geometry, the effects of $residualDiff$, of emplying the twostep method, of correcting the antidiffusion and of the ration $\frac{\mathrm{d}t}{\mathrm{d}x}$ are investigated.

Effect of $residualDiff$

The results for values of $residualDiff = 0.1$, 0.5, 0.9 and 1.0 are presented in Fig 3.4, where the lab-frame baryon density $\tilde{N}_B$, pressure P and flow velocity v are given. For these tests $dt = dx/2$, the twostep method is used and antidiffusion is corrected.

The parameter $residualDiff$ controls how much extra diffusion is added, by hand, to the system during the convection stage. Since this diffusion is added during the convection stage, the antidiffusion stage will attempt to remove it along with the inherent diffusion of the algorithm and the numerics. If $residualDiff = 1$, then there is no added diffusion, while, as its value is lowered more artificial diffusion is added. From the graphs it is evident, that as the extra diffusion is decreased the numerical solution gives a better approximation of the analytic result. However, if insufficient diffusion is added to the system, the numerics become unstable, viz. for $residualDiff = 1.0$ and even 0.9 the

numerical solution displays erroneous oscillations. Thus, adding extra diffusion removes these oscillations (which are caused by dispersion), but also degrades the solution. To obtain the best solution requires finding a balance between the effects of dispersion and extra diffusion.

Effect of twostep method

To show the advantage of using the twostep method (section §2.1.4), tests were run with and without this method for $dt = dx/2$, $residualDiff = 0.9$ and with the antidiffusion corrected. The results are shown in Fig 3.5. These graphs show the improved accuracy of the numerical solution when using the twostep method, especially in the vicinity of the shock front and front edge of the rarefaction wave.

Effect of $\frac{\mathrm{d}t}{\mathrm{d}x}$

The third set of tests show the effect of changing the ratio $\frac{\mathrm{d}t}{\mathrm{d}x}$, which is related to changing the cournat number†, Fig 3.6. Three values of dt were chosen for the testes; $\frac{dx}{2}$, $\frac{dx}{4}$ and $\frac{dx}{10}$. For all the tests $residualDiff = 0.9$, the twostep method was used and antidiffusion corrected.

It is evident, form the graphs, that the higher the value of $\frac{\mathrm{d}t}{\mathrm{d}x}$, the more accurate the numerical solution. Thus it is advisable to choose the highest value of $\frac{\mathrm{d}t}{\mathrm{d}x}$ such that the cournat number does not exceed $\frac{1}{2}$. A cournat number of $\frac{1}{2}$ is the upper limit for stability of the flux corrected transport routines in **FCTHydro**.

Effect of correcting the antidiffusion

Finally the fourth set of tests with the shock problem show how important it is to correct the antidiffusion. The results in Fig 3.7 were obtained with $dt = \frac{dx}{2}$, $residualDiff = 0.9$ and the use of the twostep method. When the antidiffusion is not corrected, the ripples due to dispersion near the shock front are accentuated. These ripples become unstable and the numerical solution diverges. It is very important to correct the antidiffusion when steep gradients occur in the system.

†The cournat number given by $v\frac{\mathrm{d}t}{\mathrm{d}x}$ reflects the fraction of a cell that is traversed in a single timestep

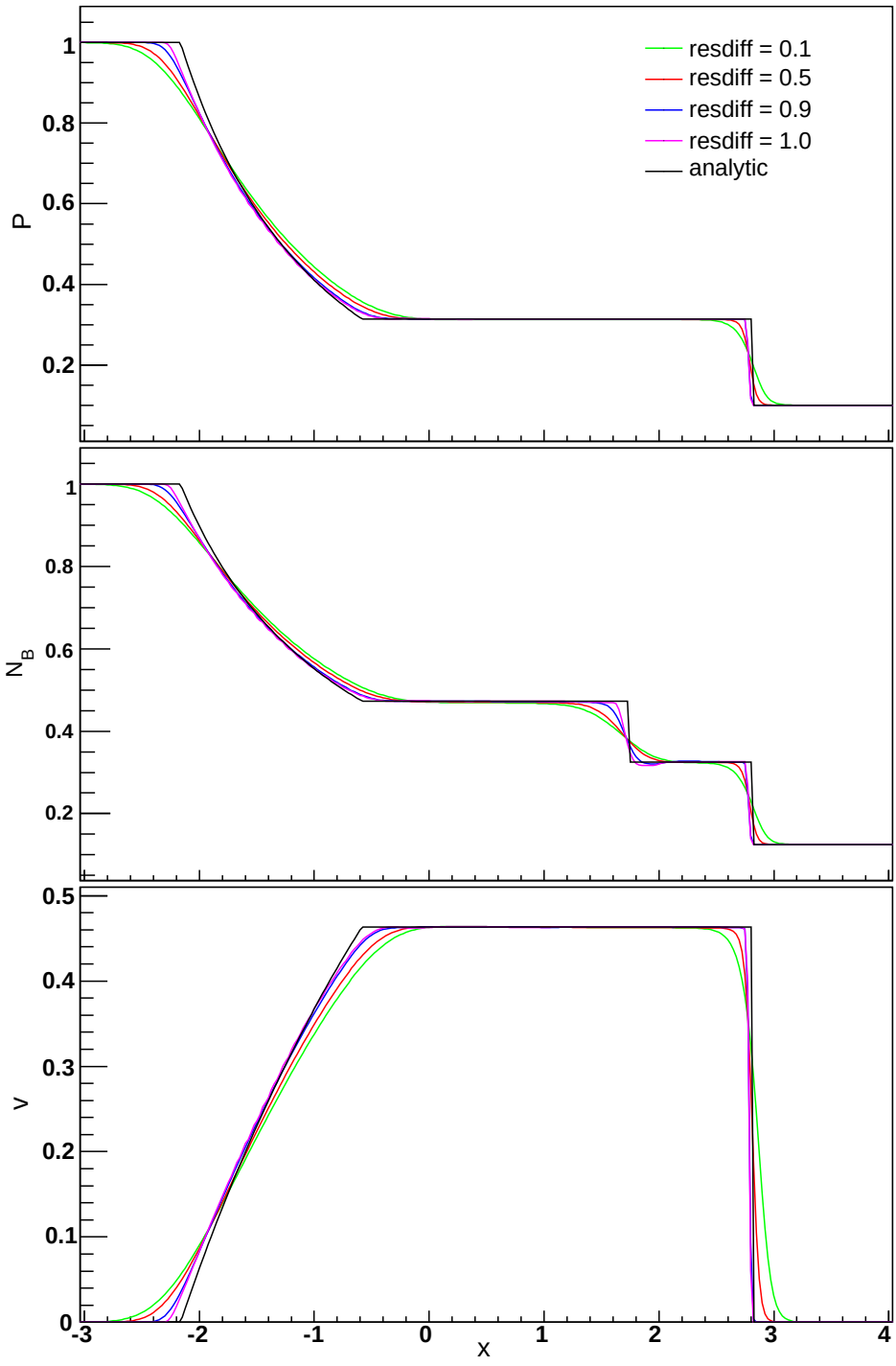


Figure 3.4 – Shock Problem: The effect of $residualDiff$ is shown by plotting the lab-frame baryon density, N_B, pressure, P, and flow velocity, v, against the analytic solution.

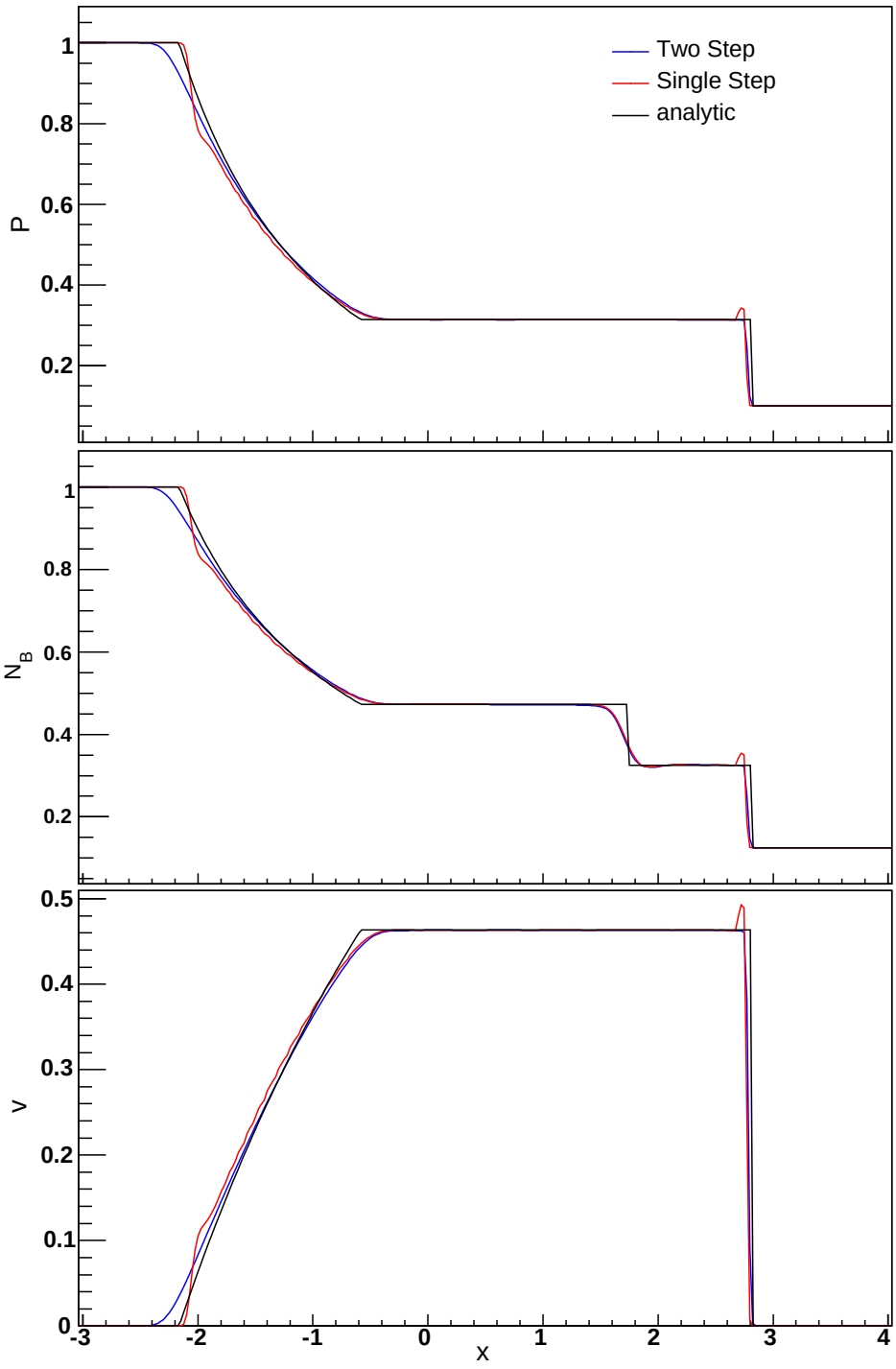


Figure 3.5 – Shock Problem: The effect of using the two step method is shown by plotting the lab-frame baryon density, N_B, pressure, P, and flow velocity, v, against the analytic solution.

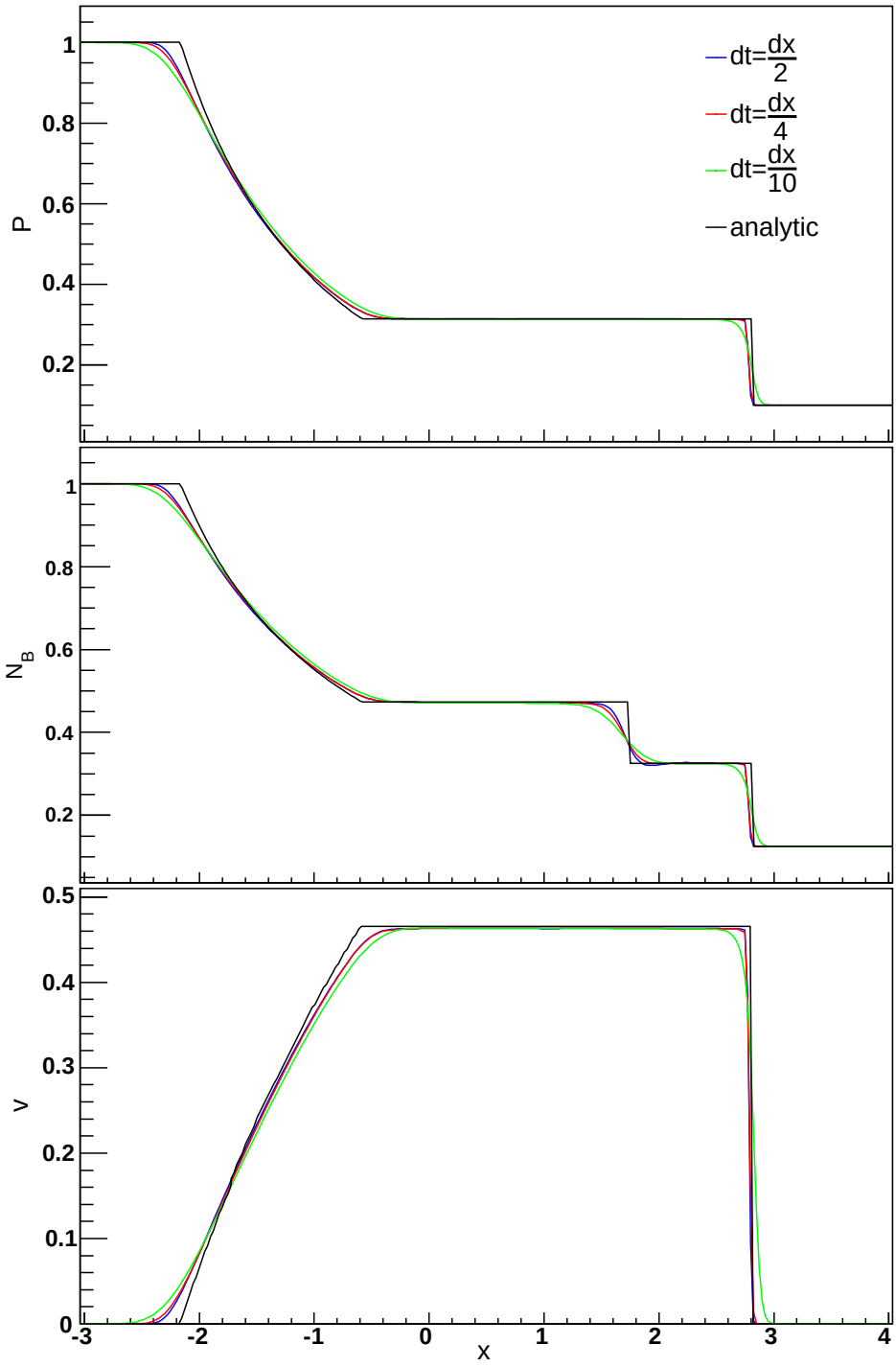


Figure 3.6 – Shock Problem: The effect of the ratio of dt to dx is shown by plotting the lab-frame baryon density, N_B, pressure, P, and flow velocity, v, against the analytic solution.

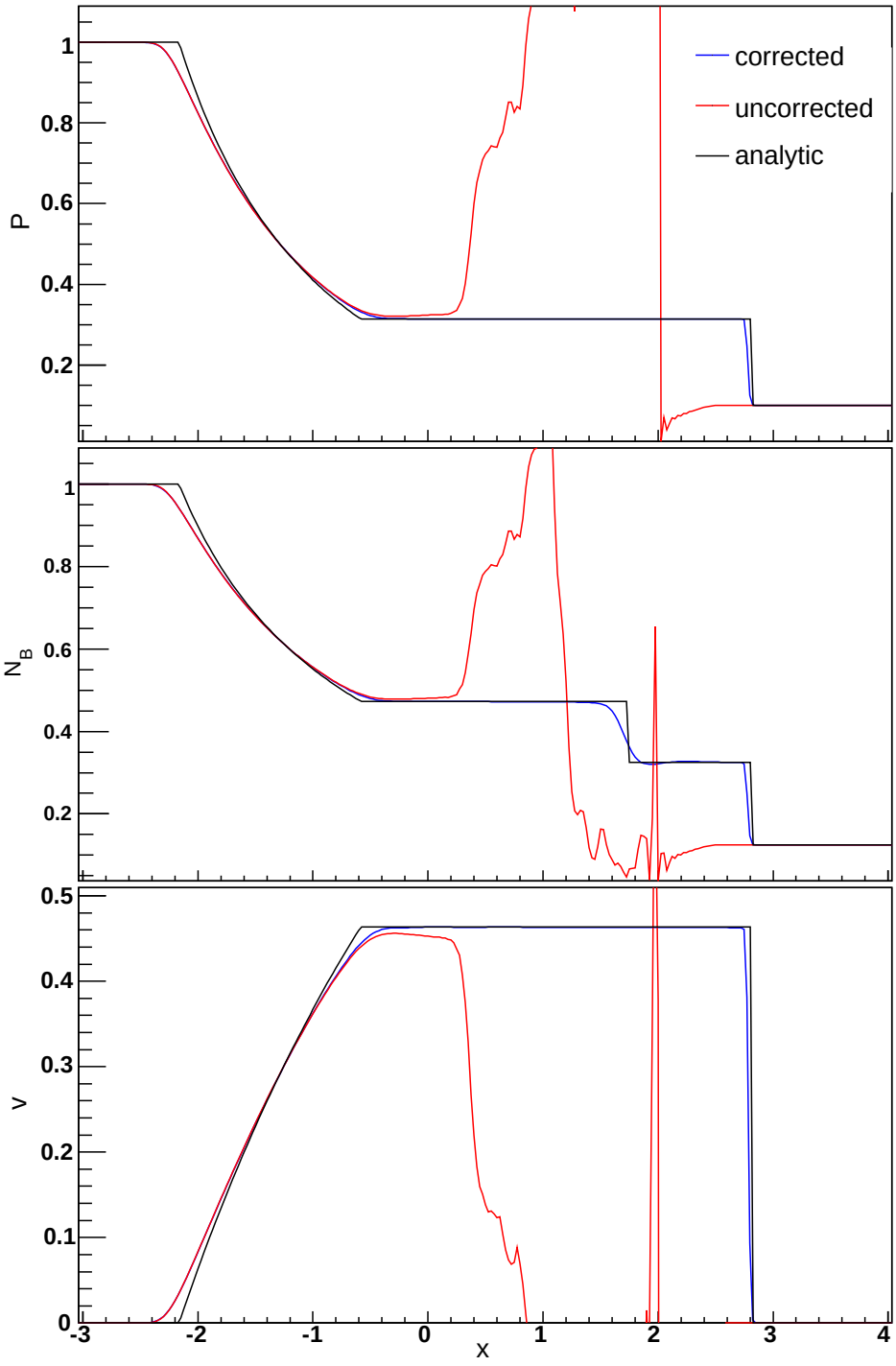


Figure 3.7 – Shock Problem: The effect of correcting the antidiffusion is shown by plotting the lab-frame baryon density, N_B, pressure, P, and flow velocity, v, against the analytic solution.

3.3.2 Landau's Solution

Analytic Solution

In [32] Landau presents an analytic solution for the expansion of matter after a nuclear collision, in the case where the matter is fully stopped during the collision. Due to the Lorentz contraction of the nuclei in the longitudinal direction, the gradients of the macroscopic variables, such as pressure, density and temperature in this direction far exceed those in the transverse direction. The result of which is that the expansion in the longitudinal direction is dominant and the system can be approximated as a (1+1) dimensional system. Landau assumed that the initial condition for his (1+1) dimensional system consisted of a static square wave for the temperature distribution, Fig 3.8, with half-width l.

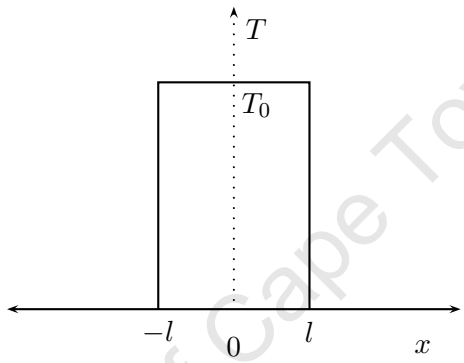


Figure 3.8 – Initial temperature distribution for Landau hydrodynamics

As the system evolves, the front edges expand outward at the speed of light, while a rarefaction wave travels inward at the speed of sound, c_s. The region between is described by simple Riemann waves. The two rarefaction waves, from the edges, collide in the centre after a time $t = l/c_s$. The region where these two waves interact defines the region of validity for Landau's analytic solution. In this text we shall refer to this as the Landau region. The geometry of this scenario is illustrated in Fig 3.9.

In finding the solution, the symmetry of the system is employed and only the right half is considered. The x coordinate is also translated such that $x = 0$ corresponds to the right edge of the distribution at $t = 0$. Under this transformation the system is symmetric about $x = -l$ and similarly $x = -2l$ corresponds to left edge of the matter. In addition, the following assumptions are made:

- The equation of state is $P = c_s^2 \varepsilon$.
- There are no chemical potentials; $\mu_B = \mu_S = \mu_Q = 0$, and thus, no baryon density.
- The system obeys ideal hydrodynamics.

The solution for the simple Riemann waves is obtained using characteristics [49] and, in terms of the new x coordinate, is

$$x = \frac{v - c_s}{1 - v c_s} t, \tag{3.21}$$

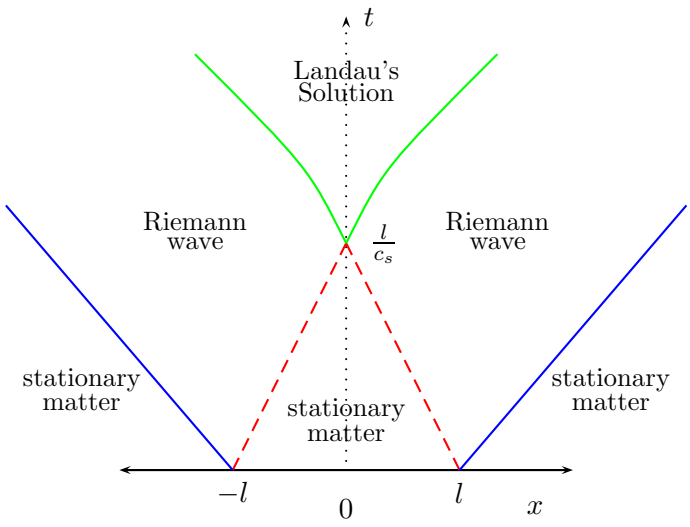


Figure 3.9 – An illustration of the different regions evident in a hydrodynamic simulation with Landau's initial conditions

where the fluid velocity, v, and temperature, T, are constant along each characteristic. On the characteristics, the flow velocity and the temperature are related via

$$\alpha c_s = w, \tag{3.22}$$

where the α is the flow rapidity, $\alpha = \operatorname{arctanh} v$, and $w = \ln \frac{T}{T_0}$.

Using entropy conservation, since the hydrodynamics is ideal, and a Legendre' transformation from the parameters $(x,t) \rightarrow (T,\alpha)$, a potential function $\chi(T,\alpha)$ is found for the Landau region. This potential function is then subject to the boundary conditions:

1. $\alpha\Big|_{(x=-l)} = 0$: which is a result of the symmetry around $x = -l$. This boundary condition states that the matter at the centre of mass is always stationary.
2. $\chi = 0$ for $\alpha c_s = -w$, this describes the boundary between the Landau region and the Riemann wave region.

With these two boundary conditions the potential takes the integral form

$$\chi(w,\alpha) = -\frac{T_o l}{c_s} e^w \int_{c_s\alpha}^{-w} e^{w'(\beta+1)} \mathrm{I}_0(\beta\sqrt{w'^2 - c_s^2\alpha^2})\mathrm{d}w', \tag{3.23}$$

where I_0 is the zeroth order Bessel function and $\beta = \dfrac{1-c_s^2}{2c_s^2}$. In terms of this potential, the x and t coordinates are then given by

$$x = \frac{e^{-w}}{T_0}\left[\frac{\partial\chi}{\partial w}\cosh\alpha - \frac{\partial\chi}{\partial\alpha}\sinh\alpha\right], \tag{3.24}$$

$$t = \frac{e^{-w}}{T_0}\left[\frac{\partial\chi}{\partial w}\sinh\alpha - \frac{\partial\chi}{\partial\alpha}\cosh\alpha\right].$$

If we now wish to implement freeze-out in this system a freeze-out criterion is required and for our purposes it is choosen as a fixed temperature $T = T_{fo}$. The freeze-out curve will then be an isotherm. With this criterion we can obtain a freeze-out curve for relativistic (1+1) dimensional hydrodynamics.

For the section of the freeze-out curve in the region described by simple Riemann waves, we know that the curve must lie on one of the characteristics as the temperature is constant along both. Thus we find the flow velocity which corresponds to the temperature T_{fo} using equation (3.22) and then the coordinates of the freeze-out curve are given by equation (3.21).

For the section of the curve in the Landau region, we fix the temperature to T_{fo} and let the flow rapidity take on values from $\alpha = 0 \rightarrow \alpha_{max} \equiv \frac{-w}{c_s}$. The coordinates are then calculated using the equations (3.24). An example of a freeze-out curve obtained from Landau hydrodynamics is given in Fig 3.10.

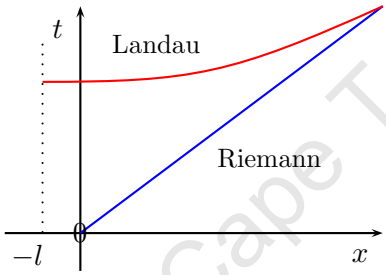


Figure 3.10 – An illustration of the right half of the freeze-out curve in a hydrodynamics system with Landau's initial conditions

Numerical Solution

As a test for the numerical accuracy, TRelHydro was used to simulate a system with Landau's initial conditions, and the results compared to the analytic solution. The massless gas equation of state was used as $P = \frac{\varepsilon}{3}$. The simulations were run with $dt = \frac{dx}{2}$, using the twostep method and using corrected antidiffusion. The value of $residualDiff$ was varied over the values 0.1, 0.5, 0.9 and 1.0. Also a grid of $N = 5001$ points was used with grid spacing $dx = 0.1$.

The initial conditions consist of a static, square wave distribution with half width $l = 1.005$fm, such that the maximum temperature is $T_0 = 1$GeV. The simulation was then run for 2600 timesteps and the freeze-out curve corresponding to temperature $T_{fo} = 0.4$GeV calculated. In Fig 3.11 the right half of the freeze-out curves are compared to Landau's analytic solution. For the comparison, the x coordinate of the analytic solution is transformed back, such that $x = 0$ corresponds to the centre of mass.

The graphs show that the numerical solution gets better as less extra diffusion is added to the system. In fact, for $residualDiff = 0.9$ the numerical solution is very accurate. It is however important to add extra diffusion during the convection stage as the simulation with no diffusion has large ripples present, Fig 3.12. Recall that extra

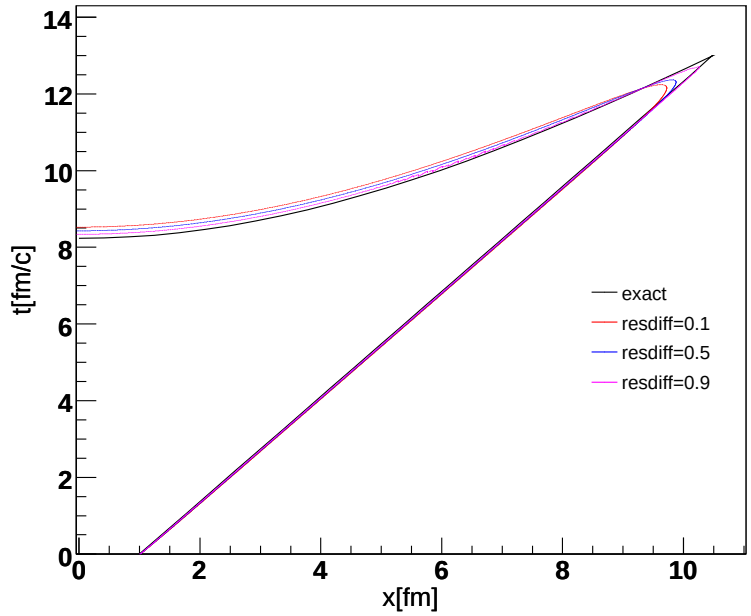


Figure 3.11 – Numerical and analytic result for the freeze-out curve in Landau hydrodynamics with $\frac{T_{fo}}{T_0} = 0.4$

diffusion added through the parameter $resdiff$ is necessary to correct for anomalous ripples caused by dispersion at a shock front.

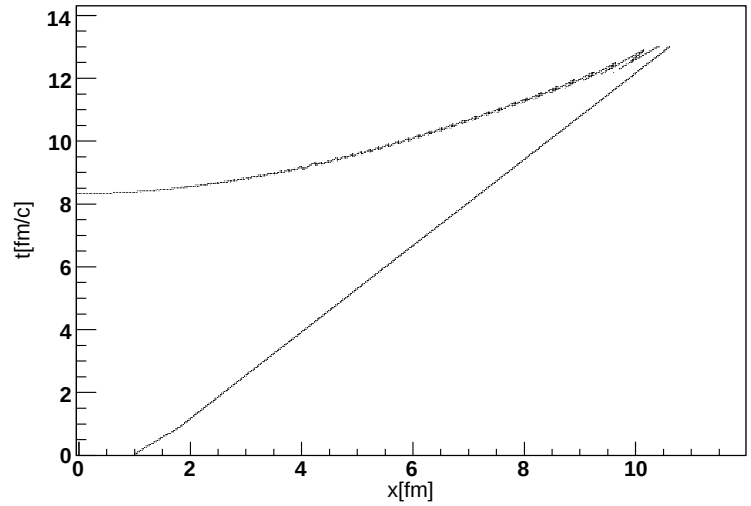


Figure 3.12 – Numerical freeze-out curve in Landau hydrodynamics with $\frac{T_{fo}}{T_0} = 0.4$ with no extra diffusion added ($resdiff = 1.0$).

Note that at the position of the centre of mass the numerical solution cools slower than the analytic solution as the freeze-out temperature is reached at a later time. This is due to a combination of the implicit numerical diffusion and the fact that the initial numerical distribution is not a pure square wave. The numerical initial condition's temperature distribution does not drop from $T = T_0$ to $T = 0$ vertically but over the width of a single grid cell, $dx = 0.1$. Thus the gradients in the numerical system are smaller than for the analytic initial condition and hence the expansion rate is slower. This in turn results in the numerical system cooling slower than the ideal system.

In support of this argument, Fig 3.13 shows that the deviation of the initial distribution away from a square wave decreases the cooling rate at the centre of mass. The figure compares the freeze-out curves for two simulations using the same parameter values where one has a square wave initial condition and the other a Wood-Saxons distribution. Both initial distributions have the same half-width 1.005fm and maxima, while the Wood-Saxons distribution has a skin depth of 0.05fm. The square wave distribution can be interpreted as a Wood-Saxons distribution with skin width zero. Again this difference

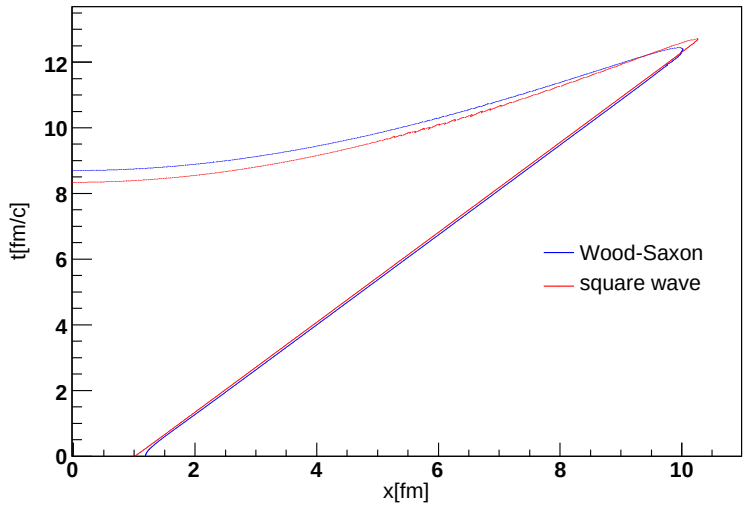


Figure 3.13 – Freeze-out curves displaying the effect on the cooling of the system due to increasing the skin width of the initial distribution.

in the cooling, as is evident at $x = 0$, is understandable as the square wave distribution has steeper gradients and thus a higher acceleration of the matter away from the centre.

At present TRelHydro can only calculate the $\frac{\mathrm{d}N}{\mathrm{d}y}$ spectra of cold matter (matter without a thermal distribution). However, we can determine the $\frac{\mathrm{d}N}{\mathrm{d}y}$ spectra for thermalised particles using the freeze-out curve obtained from the numerical solution and the Cooper-Frye freeze-out formalism. A similar method was employed in [108], where the numerical solution for the freeze-out curve from SHASTA was used to show that one cannot ignore the contribution from the spacelike[‡] (in [108] referred to as timelike)

[‡]Here we follow the convention in [46] and defined a freeze-out curve as timelike/spacelike if its normal

section of the freeze-out curve.

According to the Cooper-Frye formalism the spectrum of particles frozen out through the freeze-out hypersurface is given by

$$E\frac{\mathrm{d}N}{\mathrm{d}^3p} = \int_\sigma \mathrm{d}\sigma_\mu p^\mu f(p^\nu u_\nu). \tag{3.25}$$

For flow in the x-direction alone

$$u^\nu = \gamma(1, v, 0, 0) = (\cosh\alpha, \sinh\alpha, 0, 0)$$

and $\mathrm{d}\sigma_\nu = [-\frac{\partial x}{\partial \xi}, \frac{\partial t}{\partial \xi}, 0, 0]\,\mathrm{d}\xi$. Combining this with the form for the four momentum in terms of rapidity, y, and transverse mass, $m_\perp$, we have

$$p^\nu u_\nu = m_\perp \cosh(y-\alpha),$$
$$\mathrm{d}\sigma_\mu p^\mu = m_\perp \left[\frac{\partial t}{\partial \xi}\sinh y - \frac{\partial x}{\partial \xi}\cosh y\right],$$

and

$$\frac{\mathrm{d}N}{\mathrm{d}y} = 2\pi \int_0^\infty p_\perp m_\perp \int_\xi \left[\frac{\partial t}{\partial \xi}\sinh y - \frac{\partial x}{\partial \xi}\cosh y\right] f(m_\perp \cosh(y-\alpha))\,\mathrm{d}\xi\,\mathrm{d}p_\perp.$$

Since the flow parameters and $p_\perp$ are independent, we can switch the order of the integral, giving

$$\frac{\mathrm{d}N}{\mathrm{d}y} = 2\pi \int_\xi \left[\frac{\partial t}{\partial \xi}\sinh y - \frac{\partial x}{\partial \xi}\cosh y\right] \int_0^\infty p_\perp m_\perp f(m_\perp \cosh(y-\alpha))\,\mathrm{d}p_\perp\,\mathrm{d}\xi. \tag{3.26}$$

Assuming that the distribution function $f(p^\nu u_\nu)$ is that of massless particles which obey Boltzmann statistics, we can evaluate the inner integral

$$\int_0^\infty p_\perp m_\perp f[m_\perp \cosh(y-\alpha)]\,\mathrm{d}p_\perp = \int_0^\infty p_\perp^2 e^{-p_\perp \cosh(y-\alpha)/T}\,\mathrm{d}p_\perp \tag{3.27}$$

$$= \frac{2T^3}{\cosh^3(y-\alpha)}. \tag{3.28}$$

Thus all that is left, is to integrate along the freeze-out curve. To use the numerical solution we let $\xi = t$ and are left with

$$\frac{\mathrm{d}N}{\mathrm{d}y} = 2\pi \int_\sigma \frac{2T^3}{\cosh^3(y-\alpha)} \left[\sinh y - \frac{\partial x}{\partial t}\cosh y\right] \mathrm{d}t \tag{3.29}$$

where α and $\dfrac{\partial x}{\partial t}$ are evaluated on the freeze-out curve.

This method is valid as long as the ratio of $\dfrac{T_{fo}}{T_0}$ is not too small. This can be

is timelike/spacelike.

understood by calculating the minimum time a simulation must run and grid size which must be used when calculating the freeze-out curve for a certain $\frac{T_{fo}}{T_0}$ ratio. To determine the coordinates of the sharp point of the freeze-out curve (the furthest point from the origin, Fig 3.10) we need to evaluate equations (3.24) at the maximum flow rapidity

$$\alpha_{max} = -\frac{1}{c_s} \ln\left(\frac{T_{fo}}{T_0}\right), \tag{3.30}$$

yielding

$$x_{max} = \frac{l}{c_s} e^{\alpha_{max} c_s (\beta+1)} \left[\sinh\alpha + c_s \cosh\alpha_{max}\right], \tag{3.31}$$

$$t_{max} = \frac{l}{c_s} e^{\alpha_{max} c_s (\beta+1)} \left[\cosh\alpha + c_s \sinh\alpha_{max}\right].$$

The graphs in Fig 3.14 show x_{max} and t_{max} for $l = 1.0$ fm and $c_s = \frac{1}{\sqrt{3}}$. If the ratio $\frac{T_{fo}}{T_0} = 0.1$ then the simulation must run for the time interval 1978 fm/c on a grid of size 1973 fm. Since the initial halfwidth of the temperature distribution $l = 1.0fm$, we need dx of the order 0.01 fm for a decent approximation to the analytic initial condition. Thus we would need a grid with approximately 200,000 cells and thus at least 400,000 timestep iterations, as $dt \leq \frac{dx}{2}$[§]. This is a huge simulation and not really feasible. Also, the numerical solution may deviate fairly substantially after this number of iterations. One might suggest that since x_{max} and t_{max} are linearly dependent on l, we could just decrease l, however we would also have to decrease dx. So performing the freeze-out numerically is not really feasible for the above scenario. Thus, it is preferable to use

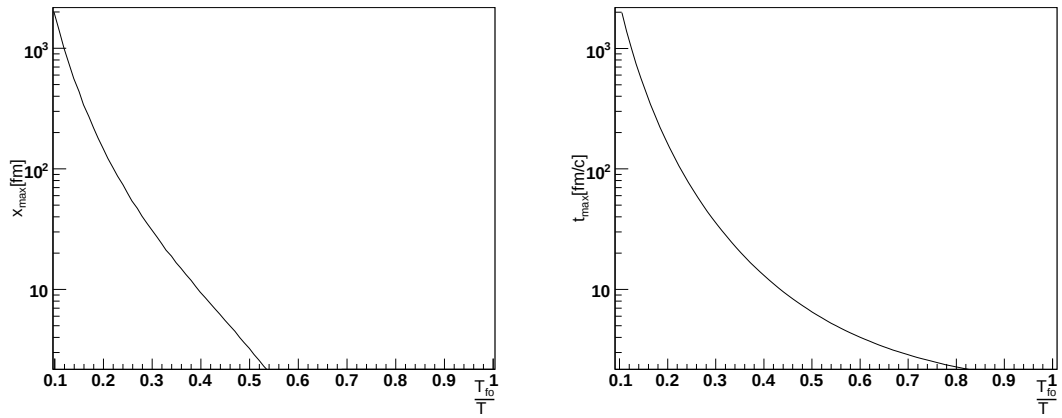


Figure 3.14 – Plots of x_{max} and t_{max} as a function of $\frac{T_{fo}}{T_0}$ with $c_s = \frac{1}{\sqrt{3}}$ and $l = 1$ fm.

[§]This is beacause the courant number $\frac{vdt}{dx} \leq \frac{1}{2}$ for stability of the algorithm and the velocity of the fluid is limited $v \leq c = 1$

some other method to determine the freeze-out spectra when the ratio $\frac{T_{fo}}{T_0}$ is small, for example see Chapter 5.

3.3.3 Tabulated EOS

One must be carefull when using the tabulated equation of state. If the grid of your tabulation is too course you could introduce large numerical errors. We will consider the example of tabulating the Hadronic equation of state used in **TRHThmCFO**. We initialised the densities with Wood-Saxon distributions with half-width 1 fm, skin-width 0.01 fm and maximum energy and baryon densities 0.9GeV fm^{-3} and 0.2fm^{-3}, respectively. We then used the equation of state classes **TRHTabEOS** and **TRHThmCFO** to determine the pressure distribution and compared the results. In Fig 3.15, we present the ratio $\frac{P}{\varepsilon}$ from **TRHThmCFO** and using two different **TRHTabEOS** objects, which were populated using **TRHThmCFO**, with different granularities. In regions where the

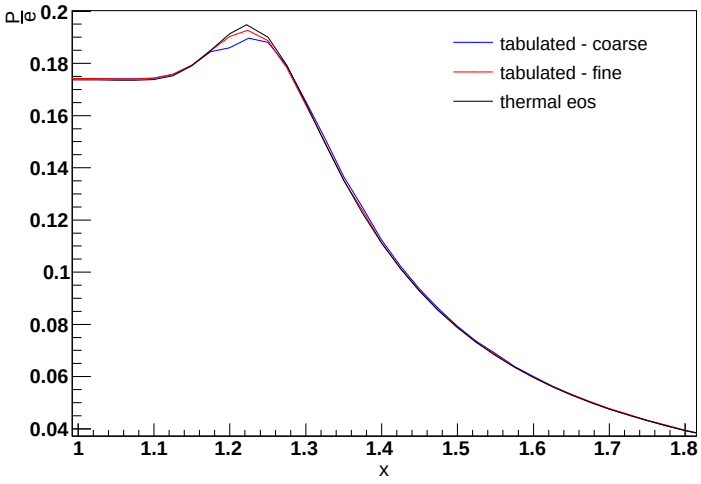


Figure 3.15 – $\frac{P}{\varepsilon}$ obtained from **TRHThmCFO** and two different **TRHTabEOS** objects which were populated using **TRHThmCFO**.

gradients changed most rapidly the courser tabulated equation of state gave a poorer approximation to the pressure values.

In describing the tabulated grids we use the following notataion; $a0 :: da :: a1$ signifies a vector starting at $a0$ with increment da between elements and ending at $a1$. For both grids, the energy density values start at 0 increase with step size 10^{-21} till 10^{-20} then in step size 10^{-20} until 10^{-19}, etc, until 0.01, where after it increases in step size 0.01 up to 1.0;

$$\varepsilon = 0, 10^{-21} \times (1,2,3,4,5,6,7,8,9), 10^{-20} \times (1,2,3,4,5,6,7,8,9), ... => 0.01 :: 0.01 :: 1$$

The course grid has the baryon density grid starting at 0 increasing with step size 0.01

to 1.0;

$$n = 0.0 :: 0.01 :: 1$$

While the fine grid has

$$n = 0, 10^{-10} * (1, 3, 5, 7, 9), 10^{-9} * (1, 3, 5, 7, 9), ... => 0.001 :: 0.00495 :: 1$$

3.3.4 Antidiffusion and Relativistic Hydrodynamics

When the equation of state is of the form $p = c_s^2\varepsilon - \beta$ and equation (3.3) is used to find the energy density, one must be carefull if T^{00} is equal to or marginally larger than $\frac{2c_sT^{0x} + \beta}{1 - c_s^2}$. Since the distributions for T^{00} and T^{0x} are treated independently during their respectively anitdiffusion stages, it possible that after applying corrected antidiffusion that

$$T^{00} < \frac{2c_sT^{0x} + \beta}{1 - c_s^2}.$$

Recall that correcting antidiffusion means that antidiffusion is only applied to situations where four consecutive points produce a monotonic line (Fig 2.2). So, if for four consecutive points of the T^{00} distribution are monotonic while the corresponding points on T^{0x} are not, then antidiffusion is applied to T^{00} but not T^{0x}. It is possible then, if T^{00} is very close to $\frac{2c_sT^{0x} + \beta}{1 - c_s^2}$, that the antidiffusion pushes T^{00}'s value below that of $\frac{2c_sT^{0x} + \beta}{1 - c_s^2}$ at some point. This would result in the square root in equation (3.3) returning a Nan in the numerics and the simulation breaking. To account for this the numerics check to see if $T^{00} < \frac{2c_sT^{0x} + \beta}{1 - c_s^2}$ after antidiffusion, and if it is it resets the values of T^{00} and T^{0x};

$$T^{00}_{new} = \frac{1}{2}\left(T^{00} + \frac{2c_sT^{0x} + \beta}{1 - c_s^2}\right), \tag{3.32}$$

$$T^{0x}_{new} = \frac{(1 - c_s^2)T^{00}_{new} - \beta}{2c_s}. \tag{3.33}$$

Chapter 4

TNonideal_xy - 2+1D

In non-equilibrium systems, the presence of the thermodynamic irreversible processes internal friction (viscosity) and thermal conductivity result in energy dissipation. When dissipation is present, ideal hydrodynamics no longer applies and we have entered the arena of non-ideal (viscous) hydrodynamics.

In this chapter we use **FCTHydro** to simulate a causal, viscous, multi-dimensional hydrodynamic system. The simulation provides some understanding of causal non-ideal hydrodynamics and is also an illustration of **FCTHydro**'s wide functionality, in that it can even be appiled to non-ideal systems. It also illustrates the application of **FCTHydro** to a multi-dimensional system, via the time-split method (section §2.1.5).

The chapter begins with a review of relativistic, causal, viscous hydrodynamics. Then an assumption is made on the form of the distributions to reduce the full (3+1) dimensional to an effective (2+1) dimensional system which is considered in the weak coupling and non-relativistic limit. Then a summary of the code used to apply **FCTHydro** to this system is presented. Finally chapter concludes with some specific applications and testing.

4.1 Theory

The well known Navier-Stokes equation, for non-relativistic non-ideal systems, was first generalised to relativistic systems by Eckart [48] and Landau and Lifshitz [49]. These non-relativistic and relativistic theories are obtained by considering first order dissipative corrections to ideal hydrodynamics. Unfortunately, because only first order corrections are considered, the equations obtained are parabolic and allow the propagation of perturbations to violate causality. In fact, the equations allow for propagations with infinite speed [109] and have many instabilities [110]. By taking second order corrections into account, a causal theory [82] was obtained and subsequently generalised [83] to relativistic systems. The equations obtained in [83] are consistent, in the non-relativistic limit, with the linearised form of dynamics equations obtained using Grad's 13-moment method [111]. The dynamical equations obtained via these methods are hyperbolic and thus causal. In the literature is common for the acausal theories to be referred to as first order theories, while the casual ones as second order theories.

For both ideal and non-ideal hydrodynamics, the dynamics are described by con-

servation equations for the conserved particle current and the energy-momentum stress tensor. The ideal systems (in 3+1 dimensions) are characterised by only five independent dynamical variables; the energy density, conserved particle density and the three components of the flow velocity. However, the non-ideal systems are characterised by *fourteen* independent variables. For both types of fluids, the systems of equations describing the dynamics are closed through the equation of state.

For our discussion of relativistic, causal viscous systems, we choose the Eckart frame [48], where the flow 4-velocity follows the conserved particle flow

$$u^\mu = \frac{N^\mu}{\sqrt{N^\nu N_\nu}}. \tag{4.1}$$

Thus a non-vanishing conserved particle current is needed. If, however, there is no conserved particle current then the Landau frame [49] must be used, where the 4-velocity follows the energy flow. Also, we assume that the fluid is close to equilibrium and that in the chosen frame only the pressure deviates from the equilibrium value. The local rest-frame energy density, ε, and conserved number density, n, attain their equilibrium values; $\varepsilon = \varepsilon_{eq}$ and $n = n_{eq}$. While, $\mathcal{P} = P_{eq} + \Pi$, with P the isotropic pressure and Π the bulk viscous pressure.

The conservation equations which govern the motion are then

$$\partial_\mu N^\mu = 0, \qquad \partial_\mu T^{\mu\nu} = 0, \tag{4.2}$$

where

$$\begin{aligned} N^\mu &= nu^\mu, \\ T^{\mu\nu} &= (\varepsilon + P + \Pi)u^\mu u^\nu - (P + \Pi)g^{\mu\nu} + 2q^{(\mu}u^{\nu)} + \pi^{\mu\nu}. \end{aligned} \tag{4.3}$$

Here, $\pi^{\mu\nu}$ is the shear stress tensor and q^ν is the heat flux. The round brackets around the indices represent symmetrisation, $A^{(\mu\nu)} = \frac{1}{2}\left[A^{\mu\nu} + A^{\nu\mu}\right]$, and the convention $g^{\mu\nu} = \mathrm{diag}(1, -1, -1, -1)$ has been chosen.

Unlike ideal systems, the entropy 4-current is no longer conserved:

$$\partial_\mu S^\mu \geq 0, \tag{4.4}$$

where

$$S^\mu = su^\mu + \frac{q^\mu}{T} + \frac{u^\mu}{2T}\left(\beta_0\Pi^2 - \beta_1 q^\nu q_\nu + \beta_2\pi^{\lambda\nu}\pi_{\lambda\nu}\right) - \frac{1}{T}\left(\alpha_0 q^\mu\Pi - \alpha_1 q_\nu\pi^{\mu\nu}\right), \tag{4.5}$$

and T is the temperature. Note the coefficients α_i and β_i that appear in the terms which are second order in dissipative fluxes. They are determined by the equation of state [112].

This causal theory depends on two assumptions. Firstly the dissipative fluxes, heat flux and viscous pressures are independent variables. Secondly, the entropy is maximum at equilibrium and its production rate is positive. Thus, the bulk pressure, heat flux and

shear stress tensor obey [112]

$$\begin{aligned}
\tau_{\Pi}\dot{\Pi} + \Pi &= -\zeta\Theta - \frac{1}{2}\zeta T\Pi\partial_{\mu}\left(\frac{\tau_{\Pi}u^{\mu}}{\zeta T}\right) + l_{\Pi_q}\nabla_{\mu}q^{\mu}, \\
\tau_q\Delta^{\mu}_{\nu}\dot{q}^{\nu} + q^{\mu} &= \kappa T\left(\frac{\nabla^{\mu}T}{T} - a^{\mu}\right) + \frac{1}{2}\kappa T^2 q^{\mu}\partial_{\nu}\left(\frac{\tau_q u^{\nu}}{\kappa T^2}\right) \\
&\quad -l_{q\pi}\nabla_{\nu}\pi^{\mu\nu} - l_{q\Pi}\nabla^{\nu}\Pi + \tau_q\omega^{\mu\nu}q_{\nu}, \\
\tau_{\pi}\Delta^{\mu\alpha}\Delta^{\nu\beta}\dot{\pi}_{\alpha\beta} + \pi^{\mu\nu} &= 2\eta\sigma^{\mu\nu} - \frac{1}{2}\eta T\pi^{\mu\nu}\partial_{\lambda}\left(\frac{\tau_{\pi}u^{\lambda}}{\eta T}\right) + l_{\pi q}\nabla^{\langle\mu}q^{\mu\rangle} \\
&\quad +2\tau_{\pi}\pi^{\alpha(\mu}\omega^{\nu)}_{\alpha},
\end{aligned} \tag{4.6}$$

where the variables Θ, $\omega^{\mu\nu}$, $\sigma^{\mu\nu}$ and a^{μ} are the expansion scalar, the vorticity, the shear tensor and the 4-acceleration, respectively. The coefficients κ, ζ and η are the thermal conductivity and the bulk and shear viscosity coefficients, respectively. The associated relaxation times, τ_i, and relaxation lengths for couplings, l_{ij}, are related [83, 112] to κ, ζ and η via the β_i and α_i of equation (4.5). In these equations the following notation has been employed. The projection on to the 3-space orthogonal to u^{μ} is given by $\Delta^{\mu\nu} = g^{\mu\nu} - u^{\mu}u^{\nu}$. The gradient vector $\nabla^{\mu} = \Delta^{\mu\rho}\partial_{\rho}$. The angular brackets in the indices represent the symmetrised spatial and traceless part of the tensor: $A^{<\mu\nu>} = [\frac{1}{2}(\Delta^{\mu}_{\sigma}\Delta\nu_{\rho} + \Delta^{\nu}_{\sigma}\Delta\mu_{\rho}) - \frac{1}{3}\Delta^{\mu\nu}\Delta_{\sigma\rho}]A^{\sigma\rho}$ and the overdot denotes $\dot{B} = u^{\lambda}\partial_{\lambda}B$.

In [113] these equations are considered in the weak coupling limit: there is no coupling between viscosity and heat ($\alpha_1 = \alpha_2 = 0$) and there are no vorticity couplings. Thermodynamic gradients are also assumed to be small allowing the terms with the one-half factor to be eliminated. Finally the assumption that there is no coupling between acceleration and the dissipative fluxes is made. With these assumptions the above equations reduce to

$$\begin{aligned}
\dot{\Pi} &= \frac{1}{\tau_{\Pi}}\left(\Pi_E - \Pi\right), \\
\dot{q}^{\mu} &= \frac{1}{\tau_q}\left(q^{\mu}_E - q^{\mu}\right), \\
\dot{\pi}^{\mu\nu} &= \frac{1}{\tau_{\pi}}\left(\pi^{\mu\nu}_E - \pi^{\mu\nu}\right),
\end{aligned} \tag{4.7}$$

where

$$\begin{aligned}
\Pi_E &= -\zeta\Theta, \\
q^{\mu}_E &= \kappa T\left(\frac{\nabla^{\mu}T}{T} - \dot{u}^{\mu}\right), \\
\pi^{\mu\nu}_E &= 2\eta\sigma^{\mu\nu},
\end{aligned} \tag{4.8}$$

are the standard Eckart thermodynamic fluxes and the expansion scalar and shear tensor

are

$$\begin{aligned} \Theta &= \nabla_\mu u^\mu, \\ \sigma^{\mu\nu} &= \Delta^{(\mu} u^{\nu)} - \frac{1}{3}\Delta^{\mu\nu}\nabla_\rho u^\rho. \end{aligned} \tag{4.9}$$

The transport equations (4.7) together with the continuity equations (4.2) form a system of hyperbolic partial differential equations for fourteen* (1+1+3+1+3+5) independent variables $\{n,\ \varepsilon,\ u^\mu,\ \Pi,\ q^\mu,\ \pi^{\mu\nu}\}$. The dynamical equations are closed with the equation of state, which gives the isotropic pressure, P, as a function of the local rest-frame energy density, ε, and conserved particle density, n.

4.1.1 2+1D Non-relativistic Hydrodynamics

It is important to gain some understanding of these non-ideal systems by first studying them in the non-relativistic limit. Thus, for the rest of this section we no longer work in natural units and must reintroduce the necessary factors of the speed of light, c. Also, to obtain an effective (2+1) dimensional system we assume that the distributions of all the variables are independent of the z coordinate. Thus all partial derivatives with respect to z are zero and there is no flow in the z direction.

On reintroducing the required factors of c:

$$\begin{aligned} u^\mu &= \gamma(c, v_x, v_y, v_z), \\ \partial_\mu &= (\frac{1}{c}\partial_t, \partial_x, \partial_y, \partial_z), \\ \Delta^{\mu\nu} &= g^{\mu\nu} - \frac{1}{c^2}u^\mu u^\nu, \\ T^{\mu\nu} &= \frac{1}{c^2}(\varepsilon + P + \Pi)u^\mu u^\nu - (P+\Pi)g^{\mu\nu} + \frac{2}{c^2}q^{(\mu}u^{\nu)} + \pi^{\mu\nu}, \\ q^\mu_E &= \kappa T\left(\frac{\nabla^\mu T}{T} - \frac{1}{c^2}\dot{u}^\mu\right). \end{aligned} \tag{4.10}$$

Substituting these into the the continuity equations (4.2) and realising that T^{00}, T^{0x}, T^{0y}, T^{xx}, T^{yy} and T^{xy} can be expressed in terms of each other, using equation (4.3),

*Allthough, the 4-velocity, heat flux and stress tensor have four, four and sixteen components, respectively, they only contribute three, three and five variables. This is because $u^\mu u_\mu = 1$, the stress tensor is symmetric ,$\pi^{\mu\nu} = \pi^{\nu\mu}$, and the orthogonality conditions $q^\mu u_\mu = 0$ and $\pi^{\mu\nu}u_\mu = 0$ must be satisfied.

equations (4.2) can be expressed as continuity equations for n, T^{00}, T^{0x} and T^{0y}:

$$\begin{aligned}
\partial_t n + \vec{\nabla}\cdot(n\vec{v}) &= 0,\\
\partial_t T^{00} + \vec{\nabla}\cdot(T^{00}\vec{v}) &= -\partial_x\left[(\mathcal{P}-\pi^{00})v_x + \pi^{xx}v_x + \pi^{xy}v_y - \gamma q^0\frac{v_x}{c} + \gamma q^x\right]\\
&\quad -\partial_y\left[(\mathcal{P}-\pi^{00})v_y + \pi^{xy}v_x + \pi^{yy}v_y - \gamma q^0\frac{v_y}{c} + \gamma q^y\right],\\
\frac{1}{c}\left[\partial_t T^{0x} + \vec{\nabla}\cdot(T^{0x}\vec{v})\right] &= -\partial_x\left[\mathcal{P} + \pi^{xx} - \pi^{0x}\frac{v_x}{c} - \gamma q^0\frac{v_x^2}{c^3} + \gamma q^x\frac{v_x}{c^2}\right]\\
&\quad -\partial_y\left[\pi^{xy} - \pi^{0x}\frac{v_y}{c} - \gamma q^0\frac{v_x v_y}{c^3} + \gamma q^y\frac{v_x}{c^2}\right],\\
\frac{1}{c}\left[\partial_t T^{0y} + \vec{\nabla}\cdot(T^{0y}\vec{v})\right] &= -\partial_x\left[\pi^{xy} - \pi^{0y}\frac{v_x}{c} - \gamma q^0\frac{v_x v_y}{c^3} + \gamma q^x\frac{v_y}{c^2}\right]\\
&\quad -\partial_y\left[\mathcal{P} + \pi^{yy} - \pi^{0y}\frac{v_y}{c} - \gamma q^0\frac{v_y^2}{c^3} + \gamma q^y\frac{v_y}{c^2}\right],
\end{aligned} \tag{4.11}$$

where we have introduced the notation $\vec{v} = (v_x, v_y)$ and $\vec{\nabla} = (\partial_x, \partial_y)$. To obtain these equations we also made use of the orthogonality conditions $\pi^{\mu\nu}u_\nu = 0$ and $q^\mu u^\mu = 0$ to express π^{0x} and π^{0y} in terms of π^{xx}, π^{yy} and π^{xy}. Similarly, the transport equations for the bulk viscous pressure and spatial parts of the stress tensor become

$$\begin{aligned}
\partial_t \Pi + \vec{v}\cdot\vec{\nabla}\Pi &= \frac{1}{\gamma\tau_\Pi}(\zeta\Theta - \Pi),\\
\partial_t \pi^{xx} + \vec{v}\cdot\vec{\nabla}\pi^{xx} &= \frac{1}{\gamma\tau_\pi}(2\eta\sigma^{xx} - \pi^{xx}),\\
\partial_t \pi^{yy} + \vec{v}\cdot\vec{\nabla}\pi^{yy} &= \frac{1}{\gamma\tau_\pi}(2\eta\sigma^{yy} - \pi^{yy}),\\
\partial_t \pi^{zz} + \vec{v}\cdot\vec{\nabla}\pi^{zz} &= \frac{1}{\gamma\tau_\pi}(2\eta\sigma^{zz} - \pi^{zz}),\\
\partial_t \pi^{xy} + \vec{v}\cdot\vec{\nabla}\pi^{xy} &= \frac{1}{\gamma\tau_\pi}(2\eta\sigma^{xy} - \pi^{xy}),
\end{aligned} \tag{4.12}$$

where

$$\begin{aligned}
\Theta &= \partial_t\gamma + \vec{\nabla}\cdot(\gamma\vec{v}),\\
\sigma^{xx} &= -\partial_x(\gamma v_x) + \frac{1}{3}\Theta + \frac{\gamma^2 v_x}{c^2}\left[\frac{v_x}{3}\Theta - [\partial_t + \vec{v}\cdot\vec{\nabla}](\gamma v_x)\right],\\
\sigma^{yy} &= -\partial_y(\gamma v_y) + \frac{1}{3}\Theta + \frac{\gamma^2 v_y}{c^2}\left[\frac{v_y}{3}\Theta - [\partial_t + \vec{v}\cdot\vec{\nabla}](\gamma v_y)\right],\\
\sigma^{zz} &= -\partial_y(\gamma v_z) + \frac{1}{3}\Theta + \frac{\gamma^2 v_z}{c^2}\left[\frac{v_z}{3}\Theta - [\partial_t + \vec{v}\cdot\vec{\nabla}](\gamma v_z)\right],
\end{aligned}$$

$$\sigma^{xx} = \frac{1}{2}\left[-\partial_x(\gamma v_y) - \partial_y(\gamma v_x)\right] + \frac{\gamma^2 v_x v_y}{6c^2}\Theta$$
$$-\frac{\gamma^2 v_x}{2c^2}[\partial_t + \vec{v}\cdot\vec{\nabla}](\gamma v_y) - \frac{\gamma^2 v_y}{2c^2}[\partial_t + \vec{v}\cdot\vec{\nabla}](\gamma v_x). \tag{4.13}$$

The transport equations for the spatial parts of the heat flux are

$$q^x = \kappa\left[-\partial_x T - \frac{v_x}{c^2}\Theta - \frac{\gamma T}{c^2}[\partial_t + \vec{v}\cdot\vec{\nabla}]v_x\right],$$
$$q^y = \kappa\left[-\partial_y T - \frac{v_y}{c^2}\Theta - \frac{\gamma T}{c^2}[\partial_t + \vec{v}\cdot\vec{\nabla}]v_y\right]. \tag{4.14}$$

The non-relativistic limit is obtained by taking the limit $c \to \infty$. In order to take this limit of the conservation equations (4.11), we must realise that $T^{00} = E$, $T^{0x} = cM^x$ and $T^{0y} = cM^y$, where E is the energy density and M^x and M^y are the momentum densities in the x and y directions, respectively. Futher more, we will consider the mass density $\rho = mn$, where m is the mass of the particles in the gas and n is the particle number density, which is a conserved quantity. Then equations (4.11) reduce to

$$\partial_t \rho + \vec{\nabla}\cdot(\rho\vec{v}) = 0,$$
$$\partial_t M^x + \vec{\nabla}\cdot(M^x\vec{v}) = -\partial_x\left[\mathcal{P} + \pi^{xx}\right] - \partial_y\left[\pi^{xy}\right],$$
$$\partial_t M^y + \vec{\nabla}\cdot(M^y\vec{v}) = -\partial_x\left[\pi^{xy}\right] - \partial_y\left[\mathcal{P} + \pi^{yy}\right],$$
$$\partial_t E + \vec{\nabla}\cdot(E\vec{v}) = -\partial_x\left[(\mathcal{P} + \pi^{xx})v_x + \pi^{xy}v_y + q^x\right]$$
$$-\partial_y\left[(\mathcal{P} + \pi^{yy})v_y + \pi^{xy}v_x + q^y\right]. \tag{4.15}$$

The equation of state for a non-relativistic gas is $\varepsilon = \frac{P}{\Gamma - 1}$, with Γ the gas constant, and the local restframe energy density is obtained via

$$\varepsilon = E + \frac{(M^x)^2 + (M^y)^2}{2\rho}. \tag{4.16}$$

Similarly the transport equations for the bulk viscous pressure and the stress tensor (4.12) reduce to

$$\partial_t \Pi + \vec{v}\cdot\vec{\nabla}\Pi = -\frac{\zeta}{\tau_\Pi}\vec{\nabla}\cdot\vec{v} - \frac{\Pi}{\tau_\Pi},$$
$$\partial_t \pi^{xx} + \vec{v}\cdot\vec{\nabla}\pi^{xx} = -\frac{2\eta}{\tau_\pi}\left[\partial_x v_x - \frac{1}{3}\vec{\nabla}\cdot\vec{v}\right] - \frac{\pi^{xx}}{\tau_\pi},$$
$$\partial_t \pi^{yy} + \vec{v}\cdot\vec{\nabla}\pi^{yy} = -\frac{2\eta}{\tau_\pi}\left[\partial_y v_y - \frac{1}{3}\vec{\nabla}\cdot\vec{v}\right] - \frac{\pi^{yy}}{\tau_\pi},$$

$$\partial_t \pi^{zz} + \vec{v}\cdot\vec{\nabla}\pi^{zz} = \frac{2\eta}{3\tau_\pi}\vec{\nabla}\cdot\vec{v} - \frac{\pi^{zz}}{\tau_\pi},$$
$$\partial_t \pi^{xy} + \vec{v}\cdot\vec{\nabla}\pi^{xy} = -\frac{\eta}{\tau_\pi}\left[\partial_x v_y + \partial_y v_x\right] - \frac{\pi^{xy}}{\tau_\pi}, \tag{4.17}$$

and the heat flux equations (4.14) to

$$\partial_t q^x + \vec{v}\cdot\vec{\nabla}q^x = -\frac{\kappa}{\tau_q}[\partial_x T + q^x]$$
$$\partial_t q^y + \vec{v}\cdot\vec{\nabla}q^y = -\frac{\kappa}{\tau_q}[\partial_y T + q^y]. \tag{4.18}$$

We have ignored the π^{0i}, π^{00} and q^0 terms as the orthogonality conditions $\pi^{\mu\nu}u_\nu = 0$ and $q^\mu u_\mu = 0$ indicate that they vanish in the non-relativistic limit. A usefull consequence of taking the non-relativistic limit is the elimination of the time derivatives which occur on the right hand side of the transport equations. This slightly simplifies the numerical simulations.

4.1.2 Initial Conditions

There are five static initial conditions built into **TNonIdeal_xy** from which to choose. These are for a rectangular Wood-Saxons', hard rectangle, hard circle, circular Gaussian and an elliptic Wood-Saxons' distribution for the energy density and baryon density. The energy density for the rectangular Wood-Saxons' is given by:

$$\varepsilon = (\varepsilon_0 - \varepsilon_4)\left(1 + \exp\left[\frac{|x| - \xi_x}{\beta_x}\right]\right)^{-1}\left(1 + \exp\left[\frac{|y| - \xi_y}{\beta_y}\right]\right)^{-1} + \varepsilon 4 \tag{4.19}$$

with half widths ξ_x and ξ_y and skin-widths β_x and β_y in the x and y directions, respectively. The energy density for the hard rectangle initial condition is given by

$$\varepsilon = \begin{cases} \varepsilon_0 & \text{if } |\mathrm{x}| \leq \xi_\mathrm{x} \text{ and } |\mathrm{y}| \leq \xi_\mathrm{y} \\ \varepsilon_4 & \text{otherwise} \end{cases} \tag{4.20}$$

for the hard circle

$$\varepsilon = \begin{cases} \varepsilon_0 & \text{if } \mathrm{x}^2 + \mathrm{y}^2 \leq \xi_\mathrm{x} \\ \varepsilon_4 & \text{otherwise} \end{cases} \tag{4.21}$$

and for the circular Gaussian

$$\varepsilon = (\varepsilon_0 - \varepsilon_4)\exp\left(-\frac{x^2 + y^2}{2\xi_x}\right) + \varepsilon_4. \tag{4.22}$$

Finally for there is an elliptic generalisation of the Wood-Saxons'

$$\varepsilon = (\varepsilon_0 - \varepsilon_4)\left(1 + \exp\left[\frac{x^2/\xi_x^2 + y^2/\xi_y^2 - 1}{\beta}\right]\right)^{-1} + \varepsilon 4. \tag{4.23}$$

In all of these ε_0 and ε_4 are the maximum and minimum values of the energy density. The baryon density has the same form as the energy density but with a maximum of n_0 and minimum of n_4. The rest of the distributions are obtained from these and the fact that the velocity $v = 0$.

4.1.3 Applying FCTHydro

Since this is a multi-dimensional system, the timestep split method (section §2.1.5) must be employed. Thus, for each of the evolution equations (4.15) and (4.17) the terms which depend on gradients with respect to x and y are separated and evolved separately. From the comparison of equations (4.15) and equation (2.2) we realise that (4.15) are conservative evolution equations and that the terms on the right hand side are the source terms. Similarly, a comparison of equations (4.17) and equation (2.24) indicates that (4.17) are non-conservative evolution equations and, again, the terms on the right are source terms. When a source term has neither explicit dependence on, nor contains derivatives with respect to x or y, then half of it is added to the x-evolution and the other half to y-evolution. To prevent a dependence on the order chosen for the one dimensional evolutions, the source terms are calculated at the beginning of each timestep.

4.2 The TNonideal_xy Code

The second application of the **FCTHydro** is for a two-dimensional non-ideal non-relativistic system where the profiles depend on the x and y coordinates. This application has a simpler setup than the previous case as it uses a simple non-relativistic gas equation of state. For this reason we have removed the implementation of a separate class for the equation of state, although this could be added relatively simply. There is also, as yet, no freezeout implementation. Thus we have only two classes, the main class **TNonideal_xy** and a container class **TNIParam**.

4.2.1 TNIParam

As with the previous implementation there is a container class for most of the parameters of the system. These are:

Nx	*Int_t	number of physical cells in x direction
Ny	*Int_t	number of physical cells in y direction
dx	*Double_t	initial size of each cell in x direction
dy	*Double_t	initial size of each cell in y direction
dt	*Double_t	timestep
hdt	*Double_t	half of timestep
$courant$	*Double_t	courant number - if negative, dt is fixed
$dtmax$	*Double_t	maximum allowed timestep
$time$	*Double_t	time of current timestep
$writeInterval$	*Int_t	number of timesteps between saving of profiles

$noWrites$	*Int_t	number of times profiles are saved
$twoStep$	*Int_t	bool switch for using twostep method (section §2.1.4)
$residualDiffTij$	*Double_t	residual diffusion coefficient for energy-momentum tensor components
$residualDiffPi$	*Double_t	residualDiff for stress tensor components
$correctAntidiffFluxTij$	*Int_t	bool switch for correcting the antidiffusive fluxes for the energy-momentum tensor components
$correctAntidiffFluxPi$	*Int_t	bool switch for correcting the antidiffusive fluxes for the stress tensor components
$Gamma$	*Double_t	Γ factor for the non-relativistic gas equation of state
$x0$	*Double_t	lower edge of x-grid
xn	*Double_t	upper edge of x-grid
$y0$	*Double_t	lower edge of y-grid
yn	*Double_t	upper edge of y-grid
$EvolvBulkPres$	*Int_t	bool switch for evolving bulk pressure
$EvolvStressTens$	*Int_t	bool switch for evolving stress tensor components
$EvolvHeatFlux$	*Int_t	bool switch for evolving heat flux components
$e0$	*Double_t	maximum, initial, local restframe energy density
$b0$	*Double_t	maximum, initial, local restframe number density
a	*Double_t	variable for skin depth of Wood-Saxon distribution
xsi_x	*Double_t	width of distribution in x direction
xsi_y	*Double_t	width of distribution in y direction
$eta0$	*Double_t	variable for adjusting the η distribution
$tau0$	*Double_t	variable for adjusting the τ_π distribution
$fred$	*Double_t	variable for adjusting the initial distributions of the stress tensor
$toll$	*Double_t	numerical vacuum tolerance
$outFileDir$	*TString	directory for data output

The variables $e0$, $b0$, a, xsi_x, xsi_y and $fred$ are used when initialising the profiles using the built in, initialisation routines (section §4.1.2). The parameter a is related to the $beta$'s in section §4.1.2; $\beta_x = a\,dx$ and $\beta_y = a\,dy$. The distributions of η and τ_π are multiplied by the factors $eta0$ and $tau0$, respectively, if $eta0, tau0 > 0$. If they are negative then η and τ_π are set to the constant values $|eta0|$ and $|tau0|$, respectively. If the value of $courant$ is negative then dt is kept constant, at its initial value, throughout the evolution. All of the variables in this class have an associated "Get" function which returns a pointer to their value. The other important function is

SetParams(ifstream $*confile$) TString

which takes as an argument the pointer to a configuration file to set the values of the variables. It returns the names of the variables which were not set in the configuration file.

4.2.2 TNonideal_xy

This is the main class for simulating the system (4.15) and (4.17). In the code we have used the notation $T00 \equiv E$, $T0x \equiv M^x$, $T0y \equiv M^y$, and $Nbnr \equiv \rho$. The 2 dimensional profiles are stored in **TMatrixD** objects with the columns corresponding to the x values and the rows to the y values. This class has a number of important members, namely:

params	**TNIParam**	container object for variable values
$T00$	***TMatrixD**	energy density
$T0x$	* **TMatrixD**	momentum in x-direction
$T0y$	***TMatrixD**	momentum in y-direction
$Nbnr$	***TMatrixD**	particle number density e
PI	***TMatrixD**	bulk stress pressure
pi_xx	***TMatrixD**	shear stress tensor component $\pi^x x$
pi_yy	***TMatrixD**	shear stress tensor component $\pi^y y$
pi_zz	***TMatrixD**	shear stress tensor component $\pi^z z$
pi_xy	***TMatrixD**	shear stress tensor component $\pi^x y$
qx	***TMatrixD**	heat flux in x-direction
qy	***TMatrixD**	heat flux in y-direction
$xvel$	***TMatrixD**	x component of flow velocity
$yvel$	***TMatrixD**	y component of flow velocity
$pres$	***TMatrixD**	isotropic pressure
$Temp$	**FCTVector**	vector for temperature profile
tau_PI	**FCTVector**	vector for τ_Π profile
tau_pi	**FCTVector**	vector for τ_π profile
tau_q	**FCTVector**	vector for τ_q profile
$lambda$	**FCTVector**	vector for λ profile
eta	**FCTVector**	vector for η profile
$zeta$	**FCTVector**	vector for ζ profile
$Fluid$	**FCTFluid**	object for doing the 1D evolutions
velocity	**FCTVeloctiy**	object for calculating the velocity dependent parameters for 1D evolutions
$xgrid$	**FCTGrid**	object for calculating the x grid geometrical parameters
$ygrid$	**FCTGrid**	object for calculating the y grid geometrical parameters
$OFNames$	*TString	array of output file names
$DataOutType$	TString	type of data output, ASCII file, histogram or matrix

DataLocation	TString	directory for data output
$D1$	**FCTVector**	D1 source profile
$D2$	**FCTVector**	D2 source profile
$D3$	**FCTVector**	D3 source profile
$C1$	**FCTVector**	C1 source profile
$C2$	**FCTVector**	C2 source profile

There are pointers to **TMatrixD** objects, which have the name of a distribution postfixed by "*o*", which stands for "old". These objects contain the distributions of the associated parameters at the begging of the timestep, e.g. $T00$ and $T00o$ are related. For the shear stress tensor components, bulk pressure and the heat flux, there are also pointers to the **TMatrixD** objects which are prefixed by "*S*". These contain values of the associated parameters which are needed for the evolution of the stress tensor, bulk pressure and heat fluxes. There are also **TH2D** objects with pointers to them, which can be used for storing snapshots of the associated distributions in a 2D histogram, e.g.. $hpixx$, is used to store the pi_xx distribution.
The are a number of member functions for this class. Some of the private functions are listed and described below:

SetSystemSize()	void
DelHists()	void
DelMatricies()	void
ConvectDensityX(Double_t *valdt*)	Int_t
ConvectDensityY(Double_t *valdt*)	Int_t
EnergyMomentumFCTX(Int_t rj)	void
StressTensFCTX(Int_t rj)	void
StressTensFCTX(Int_t rj)	void
BulkPressX(Int_t rj)	void
HeatFluxX(Int_t rj)	void
EnergyMomentumFCTY(Int_t xi)	void
StressTensFCTY(Int_t xi)	void
BulkPressY(Int_t xi)	void
HeatFluxY(Int_t xi)	void
SetSourceValues()	void
CalcDt()	void

The function SetSystemSize() creates the **TMatrixD** and **TH2D** objects on the stack and sets the size of the **FCTVector** objects, once the values of Nx and Ny have been set. DelHists() and DelMatricies() are used for releasing the memory which was used for the **TMatrixD** and **TH2D** objects. The function ConvectDensityX() is used for convecting all distributions forward one timestep *valdt* in the x-direction. ConvectDensityY() does the same in the y-direction. EnergyMomentumFCTX() evolves the rj'th row of the energy-momentum and particle distributions in the x-direction. StressTensFCTX(), BulkPressX() and HeatFluxX() do the same for the shear stress tensor components, the

bulk stress pressure and the heat flux components. Similarly the functions which are postfixed by "Y" evolve the xi'th column in the y-direction.
SetSourceValues() determines the values of the stress and heat flux parameters as well as the values of the pressure and velocity components which will be needed for calculating the source terms for the different evolution equations.
CalcDt() calculates the value of dt. It chooses the lower of $dtmax$ and the value determined by the courant number and the maximum velocity, if the courant number is set to a negative value then dt is kept at its initial value.
The functions which one must call in to simulate the system are

TNonIdeal(TString $sConfFile$)	
SetParams(TString $confFile$)	Int_t
Initialise(Bool_t $WS = 1$, Bool_t $cylinder = 0$, Bool_t $sphere = 0$, Bool_t $gaussian = 0$, Bool_t $elliptic\_ws = 0$)	void
SetOutPutTypeFile(TString $DataFilesDir$)	void
SetOutPutTypeHist(TString $DataFilesDir$)	void
SetOutPutTypeMatrix(TString $DataFilesDir$)	void
ConvectSystem(Bool_t $verbose$)	Int_t
OutputResults(Int_t k)	void
Conserve(**TMatrixD** *$density$)	Double_t

The function SetParams() takes, as its argument, the name of a configuration file. It then passes a pointer, to this file, to the necessary components for the setting of variables. This function outputs the names of the parameters not set by the configuration file to standard out. The constructor TNonIdeal() calls the function SetParams() with argument $sConfFile$. The functions SetOutPutTypeFile(), SetOutPutTypeHist and SetOutPutTypeMatrix() are used to set the format of the data output to either ASCII files or ROOT files of **TH2D**'s or **TMatrixD**'s, respectively. If they are called with the argument $DataFilesDir$, then the output files are written to directory specified by $DataFilesDir$. If there is no argument then the files are witten to the value of $DataLocation$ in the **TNIParam** object.

Initialise() is the function which initialises the grids and distributions of the system. The grids are regular in both directions with Nx cells of width dx and lower boundary $x0$ in the x-direction, and similarly for the y-direction. The arguments determine whether the distributions are going to be rectangular Wood-Saxons', hard rectangle, hard circle, circular gaussian or elliptic Wood-Saxons distributions as described in section §4.1.2. The function ConvectSystem() simulates the evolution of the system, with verbose information if $verbose$=**true**. This is the function which has the code structure to implement the twostep method if $twoStep = 1$. OutPutData() is used to output the data in the format specified above. Conserve() integrates the density distribution, pointed to by $density$, over the two-dimensional grid.

$$\text{SetParams()} \rightarrow \left\{ \begin{array}{l} \text{Initialise()} \\ \text{SetOutPutTypeFile()} \\ \quad or \\ \text{SetOutPutTypeHist()} \\ \quad or \\ \text{SetOutPutTypeMatrix()} \end{array} \right\} \rightarrow \left\{ \begin{array}{l} \text{ConvectSystem()} \\ \text{OutputResults()} \\ \text{Conserve()} \end{array} \right.$$

4.2.3 How To Use

To illustrate how one can simulate a system using **TNonideal_xy**, see the ROOT macro below.

```
gSystem->Load("../../FCTHydro/lib/libFCTHydro.so");
gSystem->Load("../lib/libNonIdeal.so");
TNonIdeal bob("conf2.file");
bob.Initialise(0,0,0,0,1);
bob.SetOutPutTypeMatrix("./datafiles/");
bob.OutputResults(0);
bob.ConvectSystem(false);
```

Firstly the **FCTHydro** library is loaded followed by the **TNonideal_xy** library. Once these libraries are loaded, a **TNonideal_xy** object, *bob*, is instantiated with the configuration file $conf2.file$ which contains the values for the parameters of the system. The distributions are then initialised with the static elliptic Wood-Saxons' distribution. Then bob is told to output snapshots of the distributions as matrices in the directory"../datafiles". The second last line tells *bob* to output the initial distributions, while the last line convects the system without displaying verbose output.

4.3 Testing and Applications

As noted in the first section of this chapter, we wish to gain an understanding of non-ideal hydrodynamic systems. To simplify matters we consider systems in the weak coupling limit, described by the equations (4.2) and (4.7). These are then simplified further by also considering the non-relativistic limit, which reduces them to the equations (4.15) and (4.15).

Consistency check

Before performing the nonideal simulations we performed a simple consistency check. The output of **TNonideal** for an ideal hydrodynamic system was compared with the output from a corresponding simulation by the LCPFCT Fortran routines. **TNonIdeal** can be used to simulate an ideal systems by setting the initial distributions of the bulk stress pressure, stress tensor and heat flux components to zero and switching off their

evolution with the switches *EvolvBulkPres*, *EvolvStressTens* and *EvolvHeatFlux*. The results for these mulit-dimensional sysems were consistent.

4.3.1 Application: Boltzmann gas of Israel particles

To study a non-ideal system we need the form for the bulk viscosity, ζ, shear viscosity, η, and thermal conductivity, κ, as well as the relaxation times τ_Π, τ_π and τ_q. One possible model is a Boltzmann gas of Israel particles. The Israel particle [114] is a relativistic generalisation of the Maxwell Molecule [115].

Introducing the reduced temperature, $z = \dfrac{mc^2}{k_B T}$, this model gives, for low temperatures ($z^{-1} << 1$) [116]

$$\begin{aligned}
\zeta &= \frac{25}{128\sqrt{\pi}} \frac{k_B T}{c\sigma} z^{-\frac{3}{2}}, \\
\kappa &= \frac{75}{128\sqrt{\pi}} \frac{c k_B}{\sigma} z^{-\frac{1}{2}}, \\
\eta &= \frac{5}{32\sqrt{\pi}} \frac{k_B T}{c\sigma} z^{\frac{1}{2}},
\end{aligned} \tag{4.24}$$

where σ is the scattering cross section. In the above expressions the expansion in z has been truncated after the highest order term. It is interesting to note that for a gas, the viscosity coefficients increase with temperature. If we now assume that the gas is ideal ($P = nk_B T$) and that the particles are hard spheres ($\sigma = r^2$, with r the radius of the spheres) then

$$\begin{aligned}
\zeta &= \frac{25}{128\sqrt{\pi}} \frac{1}{c^2 r^2 m^{3/2}} \left(\frac{P}{n}\right)^{\frac{5}{2}}, \\
\kappa &= \frac{75}{128\sqrt{\pi}} \frac{k_B}{r^2 m^{1/2}} \left(\frac{P}{n}\right)^{\frac{1}{2}}, \\
\eta &= \frac{5}{32\sqrt{\pi}} \frac{m^{1/2}}{r^2} \text{`}, \left(\frac{P}{n}\right)^{\frac{1}{2}}.
\end{aligned} \tag{4.25}$$

Next, realising that the relaxation times are related to the β_i's in equation (4.5) via [113, 116]:

$$\tau_\Pi = \zeta\beta_0, \qquad \tau_q = \kappa c^{-2} T \beta_1, \qquad \tau_\pi = 2\eta\beta_2, \tag{4.26}$$

we find that in the weakly relativistic limit ($z \to \infty$) for a Boltzmann gas they become

[83, 116]:

$$\tau_\Pi = \frac{6\zeta}{5}\frac{mc^2}{Pk_BT},$$
$$\tau_q = \frac{2\kappa}{5}\frac{m}{Pk_BT},$$
$$\tau_\pi = \frac{\eta}{P}. \tag{4.27}$$

Since in the non-relativistic limit the bulk viscosity becomes zero, there is no evolution of the bulk stress pressure. Also to simplify matters we consider a system with no heat flux. Hence there is only the evolution of the stress tensor terms.

Another piece of information required for the evolution is the gas constant Γ. By assuming a Boltzmann gas, we find in the local restframe that the pressure, $P = nk_BT$, and the energy density, $\varepsilon = \frac{3}{2}nk_BT$. Since $P = (\Gamma - 1)\varepsilon$, the gas constant $\Gamma = \frac{5}{3}$.

Numerical values

To calculate values for η and τ_π, we require the mass and radius of the particles in the gas. Let's consider the gas to consist of particles with a mass such that, at standard temperature and pressure (STP) it has the same density as air:

$$\begin{aligned} m &= \frac{\rho k_BT}{P} \\ &= \frac{(1.292\times10^{-3}\ \mathrm{g\,cm^{-3}})(1.381\times10^{-16}\ \mathrm{g\,cm^{-2}s^{-2}K^{-1}})(273.15\ \mathrm{K})}{(1.01325\times10^{6}\ \mathrm{g\,cm^{-1}s^{-2}})} \\ &= 4.810\times10^{-23}\mathrm{g}. \end{aligned} \tag{4.28}$$

To obtain the radius of the particles we will assume that the shear viscosity is that of air when at a temperature of 293.15 K: $\eta = 1.8\times10^{-4}\ \mathrm{g\,cm^{-1}s^{-1}}$ [49]. Thus, from (4.25), the square of radius of each particle is

$$\begin{aligned} r^2 &= \frac{5}{32\eta}\sqrt{\frac{mk_BT}{\pi}} \\ &= \frac{5}{(32)(1.8\ \times10^{-4})}\sqrt{\frac{(4.81\times10^{-23})(1.381\times10^{-16})(293.15)}{\pi}} \\ \therefore r &= 2.6\times10^{-8}\ \mathrm{cm}, \end{aligned} \tag{4.29}$$

and at STP the related relaxation time

$$\begin{aligned}\tau_\pi &= \frac{\eta}{P} \\ &= \frac{1.8 \times 10^{-4}\ \mathrm{g\,cm^{-1}s^{-1}}}{1.01325 \times 10^{6}\ \mathrm{g\,cm^{-1}s^{-2}}} \\ &= 1.8 \times 10^{-10}\ \mathrm{s}. \end{aligned} \tag{4.30}$$

This value sets a limit on the possible temporal step size used in the evolution. To understand why, lets assume that the flow and its gradients in the system are negligible. Then the equations for the stress tensor components (4.17) reduce to an equation of the form:

$$\partial_t \pi = \frac{\pi}{\tau_\pi}. \tag{4.31}$$

An accurate numerical simulation of this equation requires $\mathrm{d}t \ll \tau_\pi$. If the pressures and temperatures in our model are near those at STP, the simulation requires $\mathrm{d}t \sim 10^{-11}$ s.

A second constraint on $\mathrm{d}t$ is from the courant number. The stability of the **FCTHydro** routines require that the courant number, $\frac{v\,\mathrm{d}t}{\mathrm{d}x} \leq \frac{1}{2}$. To obtain another estimate for $\mathrm{d}t$, let's assume that a shock wave is created during the expansion with mach number 1.5, i.e. it travles at a speed 1.5 times the speed of sound. The speed of sound for an ideal gas

$$c_s = \sqrt{\frac{C_P k_B T}{C_V m}}, \tag{4.32}$$

where C_P and C_V are the heat capacities at constant pressure and volume, respectively. If, in addition, the gas is monotomic, then $C_P/C_V = \frac{5}{3}$ and at STP the speed of sound[†] $c_s \sim 3.6 \times 10^4\ \mathrm{cm\,s^{-1}}$. This gives the estimate for a maximum velocity of $v_{max} = 5.4 \times 10^4\ \mathrm{cm\,s^{-1}}$. If the the initial condition distributions for the hydrodynamic system are of the order of a centimetre then they would suggest a grid spacing of $\mathrm{d}x \sim 0.1 - 0.01$ cm and as such a step size of $\mathrm{d}t \sim 10^{-7}$ s. Thus the shock wave would need an highly unrealistic mach number of the order of 10^4 before the courant number determined a stepsize of the order of the one determined by the relaxation time.

The value of τ_π indicates the necessity of a much smaller value for $\mathrm{d}t$ than the courant number does and as such a much larger number of timesteps would be required to simulate a non-ideal as apposed to the associated ideal system. Ideal systems do not contain τ_π and thus $\mathrm{d}t$ only depends on the courant number. Fortunately this discrepancy will not occur when treating an ultra-relativistic gas as is shown in section §4.3.2.

Since our primary concerns are to gain an understanding of the relativistic systems and to show the validity of the application of **FCTHydro** to non-ideal systems, we need not fix ourselves to the quantitative values obtained above. In fact to show the validity of the application of **FCTHydro** we should test a variety of η and τ_π values. Similarly we are interested in the effect of τ_π on the system, as this is a parameter which does not occur in the widely studied first order theories.

There are two possible means of increasing the value of τ_π, while keeping its qualitative behaviour. The first method is to calculate η and τ_π via (4.25) and (4.27) and

[†]For diatomic molecules of which our air primarily consists of, $C_P/C_V = \frac{7}{5}$ and $c_s \sim 3.3 \times 10^4\ \mathrm{cm\,s^{-1}}$

then manually increase τ_π by a multiplying through a factor [case A]. The second is to calculate η, increase its value manually and then calculate τ_π via (4.27) [case B]. Note that we only consider increasing τ_π as decreasing it would necessitate an even smaller value of $\mathrm{d}t$.

Initial Conditions and System Setup

For all the simulations below we used the same setup and initial conditions, which are described below. Items appearing within square brackets "[ ]" indicate which variables of **TNonideal_xy** are used to create the condition and their values.

Initial Condition: A static circular Wood-Saxon's distribution with a radius of 2 cm and skin width $a = 3\,\mathrm{d}x$ [Initialise(0,0,0,0,1); $xsi\_x = 2$; $xsi\_y = 2$; $a = 3$] for the energy density T^{00} and the mass density of the gas $Nbnr$. Since it is static, the initial x and y velocities are zero as are the momenta in these two directions; T^{0x} and T^{0y}. The initial pressure is obtained using the equation of state [$Gamma = 1.\dot{6}$] and the energy density, T^{00}. The heat flux components and the bulk viscosity are initialised to zero. The components of the stress tensor are initialised as $\pi_{xx} = -0.1 * \frac{P}{6}$, $\pi_{yy} = -0.1 * \frac{P}{6}$, $\pi_{zz} = 0.1 * \frac{P}{3}$ and $\pi_{xy} = 0$ [$fred = 0.1$]. The initial distribution for the energy density has a maximum of 1.52×10^8 and minimum of 1.52×10^6 $\mathrm{g\,cm^2 s^{-2}}$ [$e0 = 1.52\mathrm{e}8$; $e4 = 1.52\mathrm{e}6$]. While the density has a maximum of 1.292×10^{-2} and a minimum of 1.292×10^{-3} $\mathrm{g\,cm^{-3}}$ [$b0 = 1.292\mathrm{e}{-2}$, $b4 = 1.292\mathrm{e}{-3}$]. These are equivalent to setting up a system with a region of gas which has a maximum pressure 100 times that of standard pressure and a temperature 10 times standard temperature which expands into a region of gas at STP.

Geometry: A symmetric grid in both the x and y directions was used. The number of grid elements in both directions was 300 [$Nx = 300$; $Ny = 300$]. A grid spacing of 0.1 cm was the same for both directions [$dx = 0.1$; $dy = 0.1$]. The step size was set to 3×10^{-7} s [$dt = 3\mathrm{e}{-7}$] and was fixed[‡] to this value by setting the courant number to a negative value [$courant < 0$]. This is needed in order compare outputs from different simulations at the same output times.

System Parameters: For the energy, density and momentum components the antidiffusive flux is corrected [$correctAntidiffFluxTij = 1$], but not for the stress tensors [$correctAntidiffFluxPi = 0$]. The tracelessness of the stress tensor is violated when the antidiffusive flux is corrected, as the antidiffusive flux is altered in different manners for the different stress tensor components as determined by the shape of their distributions. There is no extra diffusion added for the energy, density and momentum components [$residualDiffTij = 1.0$], while some is added in the evolution of the stress tensor [$residualDiffPi = 0.85$] to smoothen the evolution since its antidiffusion is not corrected. The twostep method (section §2.1.4) is employed for the evolution [$twoStep = 1$]. During the simulation the evolution of the heat flux and bulk pressure is switched off and for the stress tensor it is switched on [$EvolvHeatFlux = 0$; $EvolvBulkPres = 0$; $EvolvStressTens = 1$].

All of the data shown below was outputted after 180 time steps, which corresponds to the time $t = 5.4 \times 10^{-5}$ s.

[‡]If the courant number is set to a positive value then $\mathrm{d}t$ is calculated using the max velocity, dx and the courant number

Case A

For these simulations η and τ_π are calculated via (4.25) and (4.27) and then τ_π is increased by some factor. The increase is achived with a multiplicative factor which is set by the parameter $tau0$. With the initial conditions above and step size $\mathrm{d}t = 3 \times 10^{-7}$, it was found that **TNonideal_xy** produces results for values of $tau0 = 10^5$ and above. For values of $tau0 = 10^4$ and below the simulation dies.

With the initial conditions as they are, the initial values of the relaxation time given by (4.27): $5.5 \times 10^{-12} \leq \tau_\pi \leq 1.8 \times 10^{-10}$. Thus it is understandable that the simulations are only stable when $tau0 = 10^5$, as these are the factors necessary for $\tau_\pi > \mathrm{d}t$.

In Fig 4.1 the cross-section, through the centre, of the energy density (T^{00}) distributions for various values of $tau0$, after 180 time steps, are plotted. Note that the distributions obtained for values of $tau0 \geq 10^8$, (a), are nearly identical. Also, the distributions for 10^5 and 10^6, (b), are nearly identical and are very close that of an ideal fluid. However, Fig 4.1(c) shows that a transition occurs as $tau0$ varies between 10^6 and 10^7.

The timescales for which the initial conditions for the stress tensor affects the evolution of the system increases with increasing τ_π. This is evident from the comparison of the stress components distributions for $tau0 = 10^8$ in Fig 4.2 with those for $tau0 = 10^6$ in Fig 4.3. The distributions for $tau0 = 10^8$ have a similar form to those of the initial condition. They appear to have only been convected non-conservatively because the correspondingly higher value of τ_π has minimised the effect of the source terms on the right hand side of equation (4.17). For $tau0 = 10^6$ the distributions do not resemble the initial conditions at all.

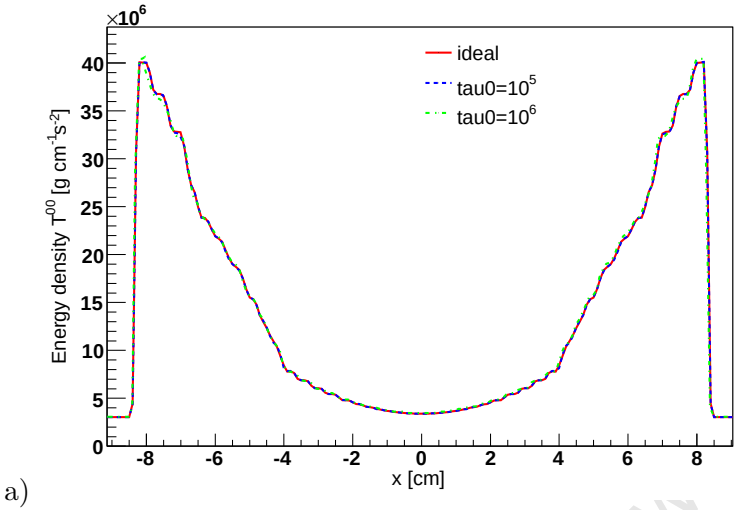
If we consider the system where the initial conditions for the stress tensor components are now reduced by a factor of 10 ($\pi_{xx} = -0.01 * \frac{P}{6}$, $\pi_{yy} = -0.01 * \frac{P}{6}$, $\pi_{zz} = 0.01 * \frac{P}{3}$ and $\pi_{xx} = 0$ [$fred = 0.01$]) the transition for $10^6 \leq tau0 \leq 10^8$ still exists. However, the distributions for 10^8 and 10^6 are closer than for the previous initial conditions.

Case B

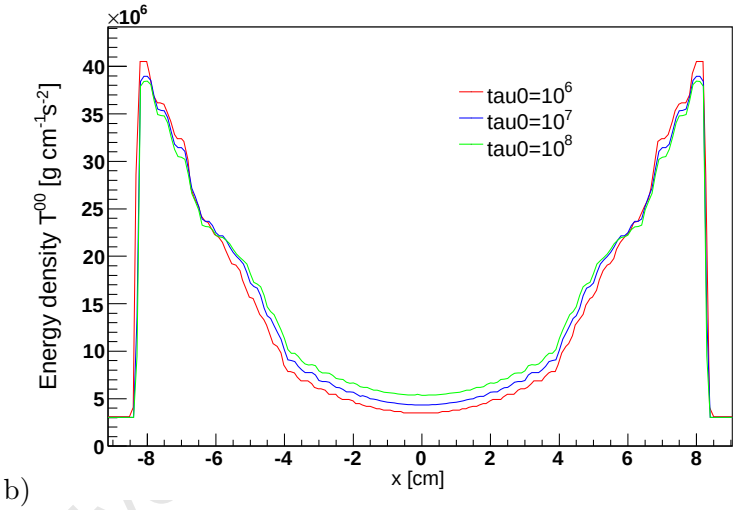
For these simulations η is calculated via (4.25) and then increase by a multiplicative factor where after τ_π is calculated via (4.27). The multiplicative factor is set by the parameter $eta0$. This has the effect of increasing both the effective viscosity and its relaxation time by a factor of $eta0$.

In Fig 4.4 the cross-section, through the centre, of the energy density (T^{00}) distributions for various values of $eta0$, after 180 time steps, are plotted. The simulations die for $eta0 \leq 10^4$. This is because, as with case A, values of $eta0 \geq 10^5$ are needed in order for the effective τ_π to be greater than $\mathrm{d}t$. The more $eta0$'s value is increased the further the solution deviates from that of the ideal system. It is evident from the graphs that the higher the effective value of the viscosity the lower the shock waves and the higher the central region; the matter addhears more to itself.

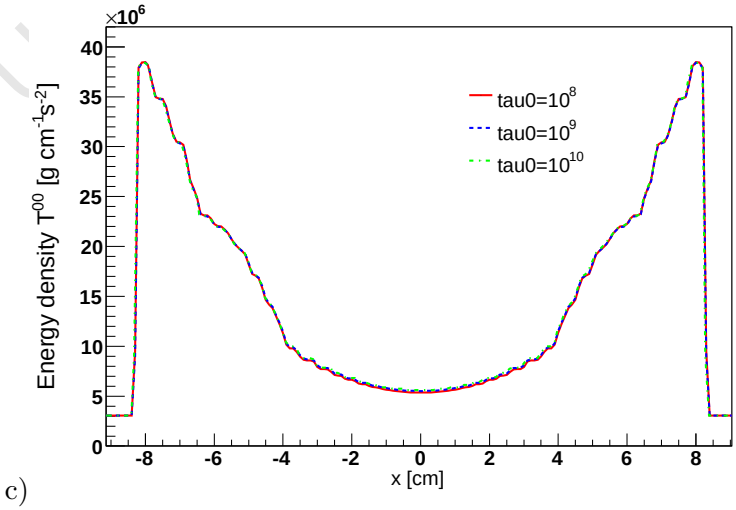
Looking at the graphs one may be concerned by fact that the viscous systems for high viscosities seem to be expanding faster than the ideal system. As the viscosity is increased the effect of the stress tensor on the evolution is increased. Recall now that the stress tensor components are evolved with extra residual diffusion to stabilise the system, as



a)



b)



c)

Figure 4.1 – The cross-section of the energy density (T^{00}) distribution after 180 time steps (5.4×10^{-5}s) for various values of the relaxation time's multiplicative factor $tau0$. In (a) the solution to the associated ideal hydrodynamic system is also plotted.

one cannot use corrected antidiffusive fluxes with the stress tensor. This added diffusion results in a slight broadening of the distributions of the stress tensor, which in turn broadens the distribution of the energy and momentum. Thus the distributions appear to travel marginally faster than they actually are.

One must be cautious when increasing the viscosity for if it is increases to much, the system initially expands but then begins to contract in the centre. This behaviour is shown in Fig 4.5 for $eta0 = 10^7$, where the cross-section of the energy density distribution is plotted at various times. Note that the central value for the last time $t = 5.40 \times 10^{-5}$ s is higher than for $t = 4.32 \times 10^{-5}$ s. This is not unreasonable as when $eta0 = 10^7$ the initial value of $\eta \sim 3 \times 10^2$ g cm^{-1}s^{-1} which is more than 5 times the viscosity of molasses at STP. Fig 4.6 shows the distributions for the stress tensor components in the case where $eta0 = 10^6$.

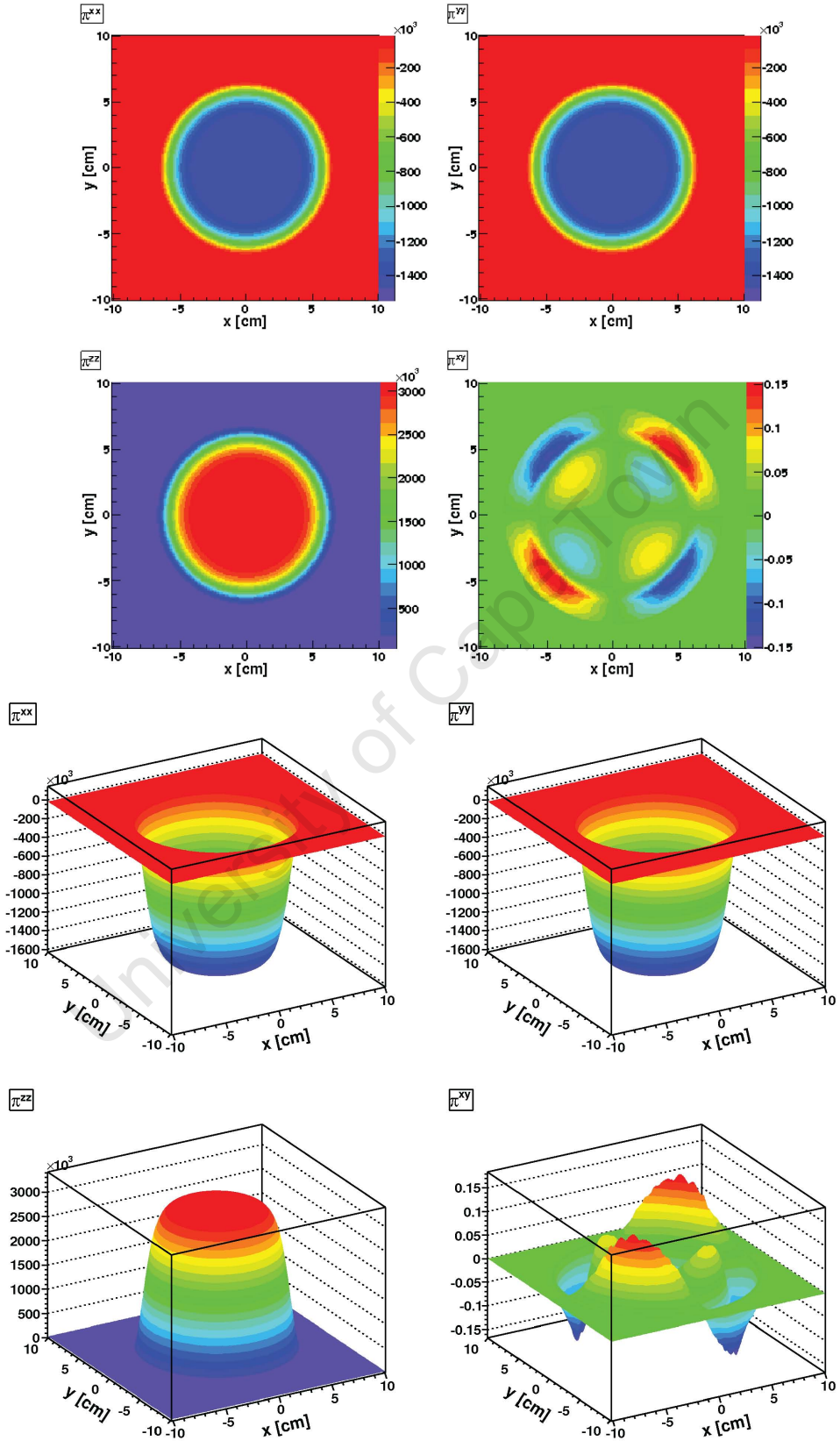


Figure 4.2 – The stress tensor components π^{ij} [g cm^{-1}s^{-2}] for $tau0 = 10^8$, after 180 time steps (5.4×10^{-5}s).

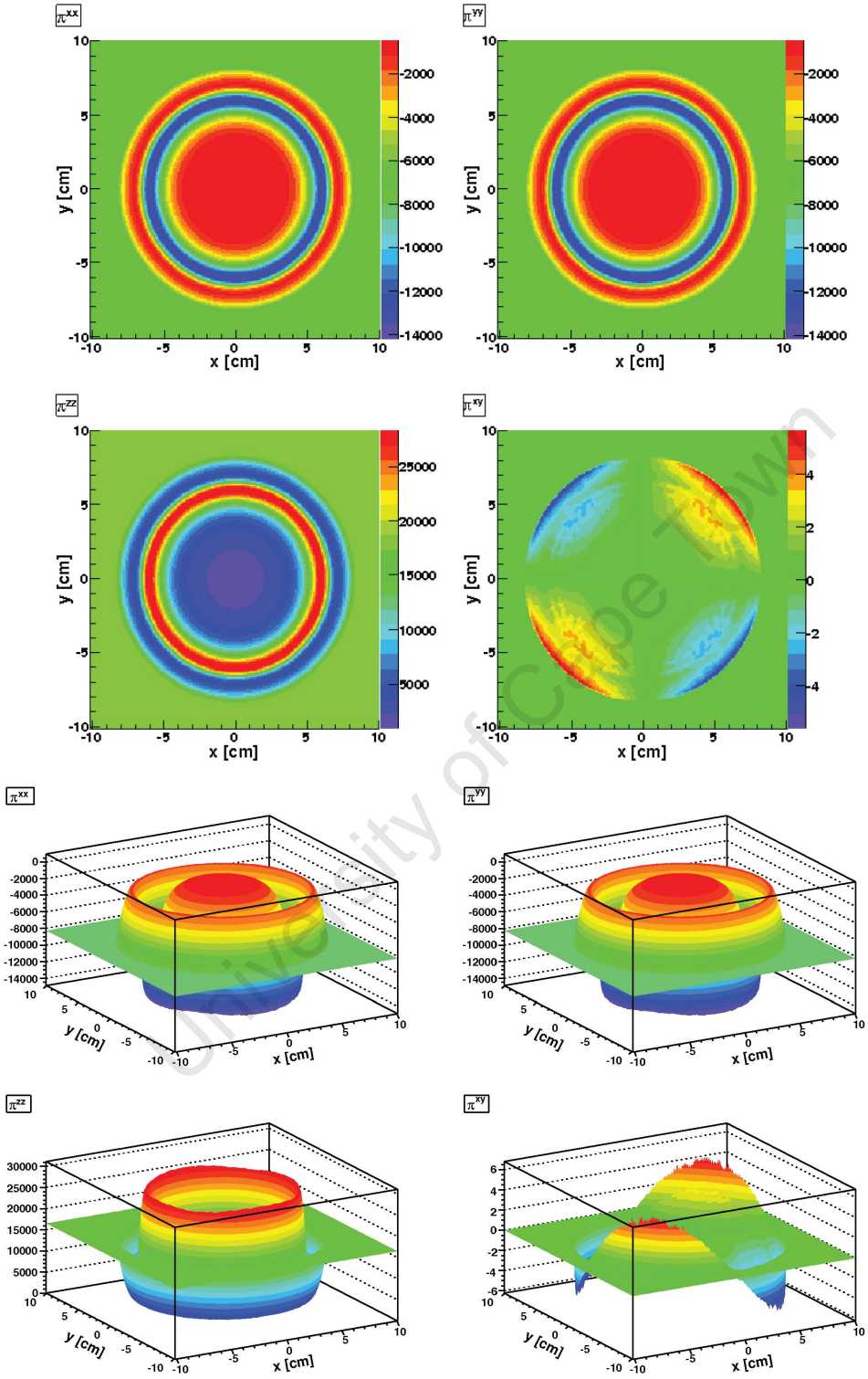


Figure 4.3 – The stress tensor components π^{ij} [g cm^{-1}s^{-2}] for $tau0 = 10^6$, after 180 time steps (5.4×10^{-5}s).

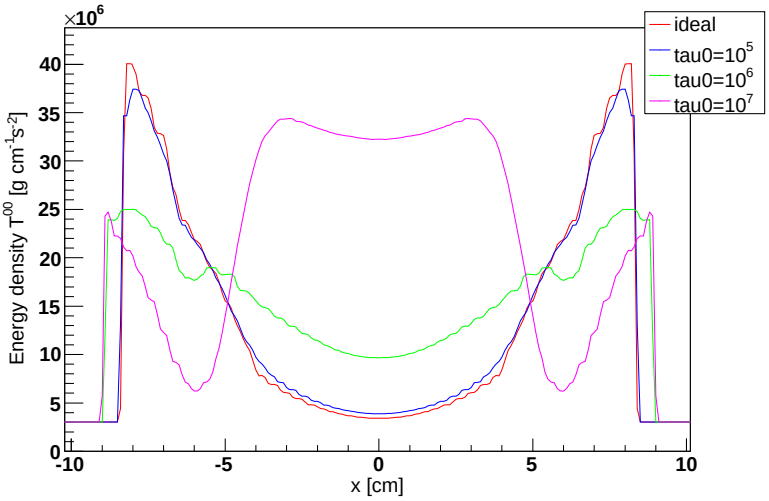
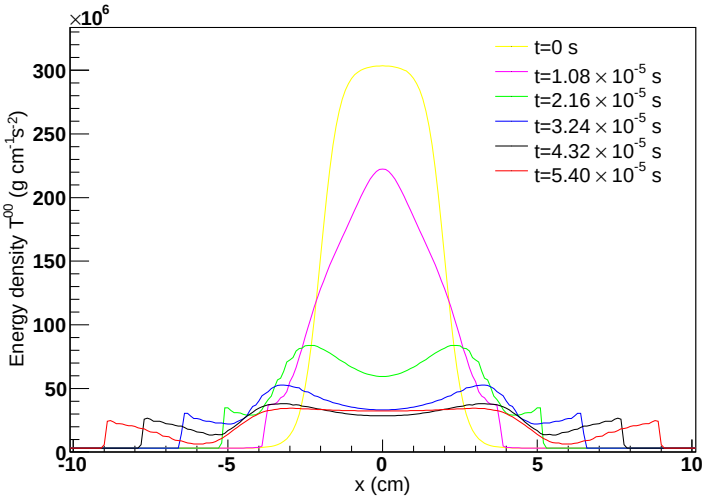


Figure 4.4 – The cross-section of the energy density (T^{00}) distribution after 180 time steps (5.4×10^{-5}s) for various values of the viscosity's multiplicative factor $eta0$. These are plotted versus the solution for the associated ideal system.



a)

Figure 4.5 – The cross-section of the energy density (T^{00}) at various times for $eta0 = 10^7$.

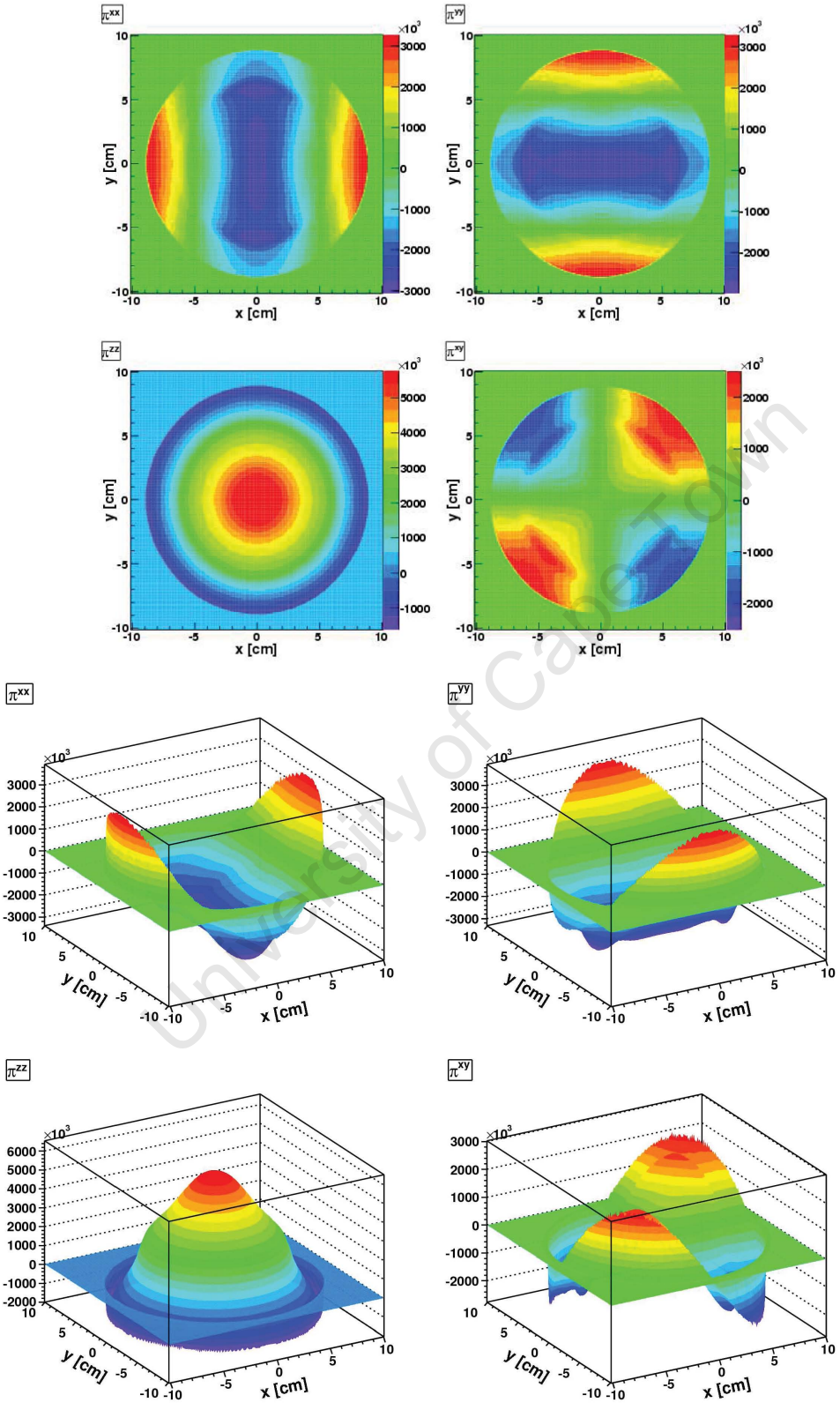


Figure 4.6 – The stress tensor components π^{ij} [g cm^{-1}s^{-2}] for $eta0 = 10^6$, after 180 time steps (5.4×10^{-5}s).

Case C

Another possible means of trying to simulate a system where the value of $\mathrm{d}t$ suggested by τ_π is smaller than that by the courant number, is to perform multiple iterations for the stress tensors components to each time iteration for the energy, density and momenta. In other words, to one iteration for the energy, density and momenta over a step size $\mathrm{d}t$, perform N iterations for the stress tensor with step size $\frac{\mathrm{d}t}{N}$. The structure of **TNonideal_xy** was altered to create **TNonideal_xy_pi10** which perform simulations in this manner[§]. It uses 10 iterations of step size $\frac{\mathrm{d}t}{10}$ to each iteration of the energy, density and momenta of step size $\mathrm{d}t$. Simulations were then run with the same setup and initial conditions as Cases A and B. The results from the original **TNonideal_xy** are compared to those of **TNonideal_xy_pi10**. In the following we use the notation of a preceding subscript 10 to indicate outputs from **TNonideal_xy_pi10**, e.g. the energy density obtained from **TNonideal_xy_pi10** is indicated as ${}_{10}T^{00}$

With the setup of, Case A, **TNonideal_xy_pi10** is able to obtain a solution for values of $tau0 \geq 10^4$. This is a factor of 10 better than **TNonideal_xy**, which is expected. With the same $\mathrm{d}t$ chosen for both systems **TNonideal_xy_pi10** uses an effective $\mathrm{d}t/10$ step-size when evolving the stress-tensor. However for the values of $tau0$ for which **TNonideal_xy** also obtains a solution, the solutions from the two simulations are nearly identical. In Fig 4.7 the ratio of the energy densities ${}_{10}T^{00}/T^{00}$ (the ratios of the solutions from **TNonideal_xy_pi10** to that from **TNonideal_xy**) are compared. The comparison is done for $tau0 = 10^5$, 10^7 and 10^8, i.e. the smallest value for which both codes produce a solution, a value which occurs within the transition region and one which occurs on other side of the transition region (see Fig 4.1, recall that the solutions for **TNonideal_xy_pi10** and **TNonideal_xy** are nearly identical). For $tau0 = 10^4$ the solution is even closer to the ideal case than for $tau0 = 10^5$. (see Case A above)

Similarly, with the setup of Case B, **TNonideal_xy_pi10** obtains a solution for values of $eta0 \geq 10^4$. Which is again a factor of 10 better than **TNonideal_xy**. For the lower value of $eta0$ for which **TNonideal_xy** also contains a solution, the solutions are nearly identical, Fig 4.9 (a) & (b). However for larger values of $eta0$ the solutions are significantly different, Fig 4.9 (c).

You will notice than in general the solutions from **TNonideal_xy_pi10** are smoother, which is not unexpected as the effective step-size for the evolution of the stress tensor is a factor of 10 smaller. The behaviour shown in Case B for values of $eta0$ which are significantly high is repreduced here; the system initially expands, but then begins to contract. This is demonstrated for $eta0 = 10^7$ in Fig 4.8, where the central value of the last output is higher than that of output before it.

[§]For a description of **TNonideal_xy_pi10**'s parameters and functions see Appendix D.

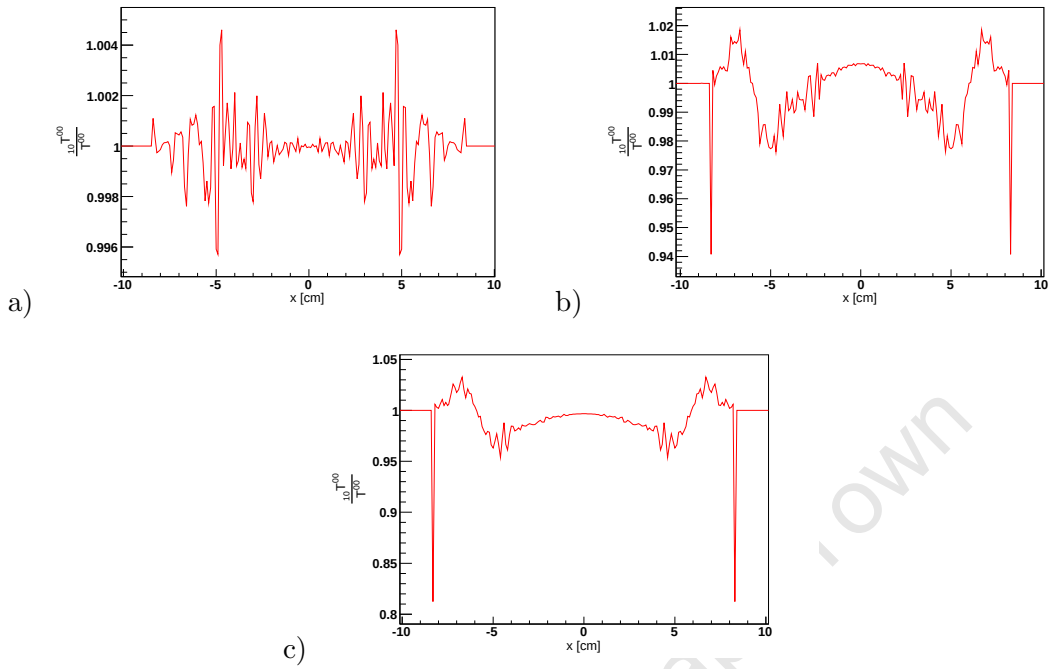


Figure 4.7 – Cross-section of the energy density ratio ${}_{10}T^{00}/T^{00}$, after 180 time steps (5.4×10^{-5}s). [The ratio of the solution from **TNonideal_xy_pi10** to that from **TNonideal_xy**.] The ratios are for $tau0 = 10^5$ (a), $tau0 = 10^7$ (b) and $tau0 = 10^8$ (c).

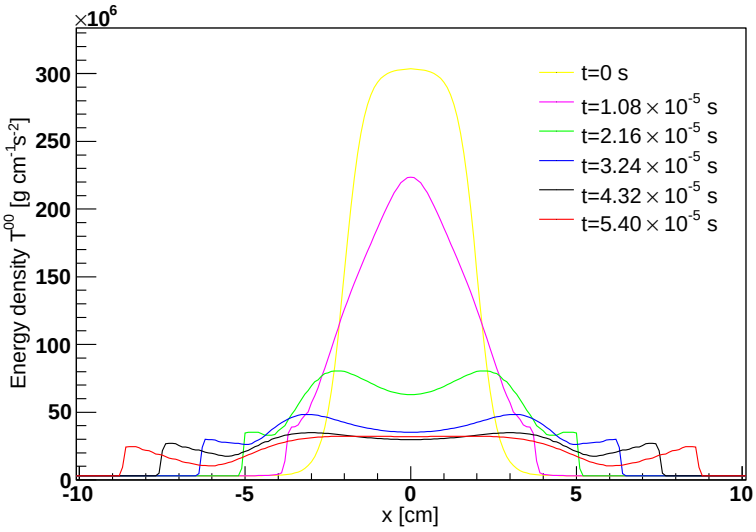


Figure 4.8 – The cross-section of the energy density (${}_{10}T^{00}$) at various times for $eta0 = 10^7$.

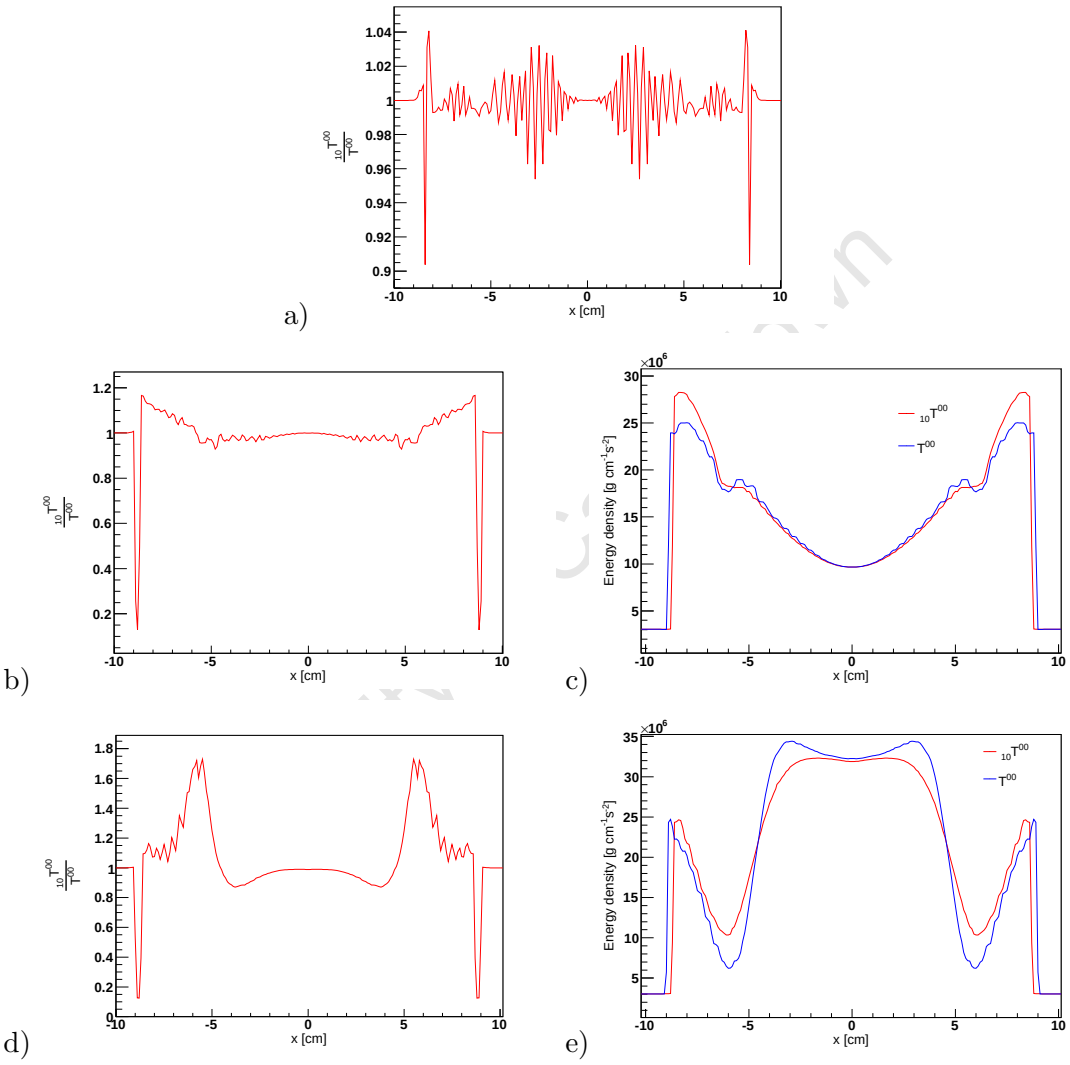


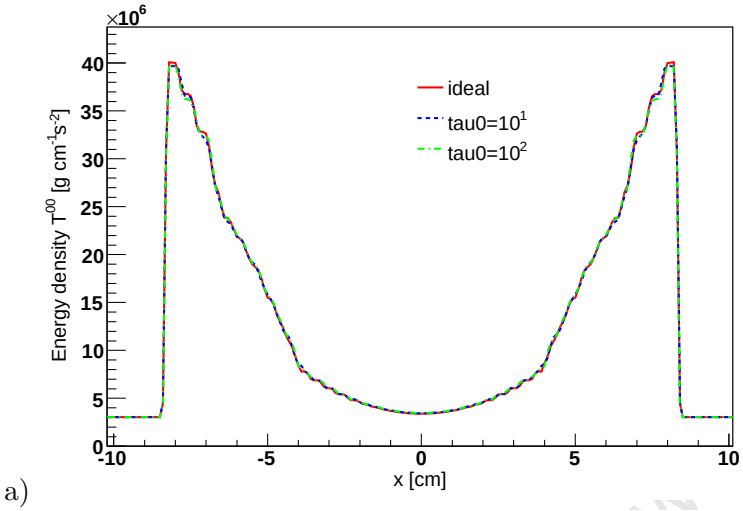
Figure 4.9 – A comparison of the energy density cross-sections obtained using **TNonideal_xy_pi10** ($_{10}T^{00}$) and **TNonideal_xy** (T^{00}), for different values of $eta0$: $eta0 = 10^5$ in fig (a), $eta0 = 10^6$ in figs (b) & (c) and $eta0 = 10^7$ in figs (d) & (e). All plots are for data obtained after 180 time steps (5.4×10^{-5}s)

Case D

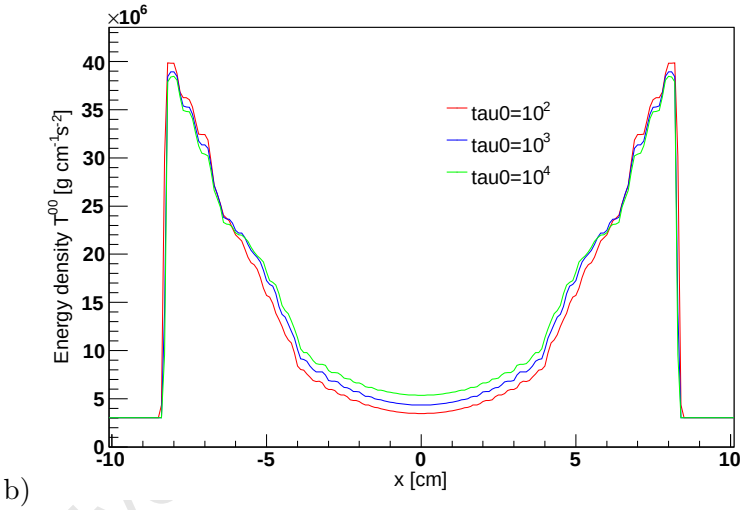
For interest we also consider what happens when $eta0 = 10^4$ and 10^6 and let $tau0$ take on a variety of values. This increases the value of η determined by (4.25) by the factor $eta0$ and the value of τ_π determined by (4.27) by the factor $eta0 \times tau0$.

Letting $eta0 = 10^4$ and using the same initial conditions and setup as used in the cases above we find that **TNonideal_xy** obtains a solution for $tau0 \geq 10^1$. Again there is a transition that occurs as $tau0$ is increased, Fig 4.10. The solutions for $tau0 = 10^1$ and 10^2 are nearly identical are are very close to the ideal systems solution. Solutions for $tau0 \geq 10^4$ are also incredibly similar and $tau0 = 10^2$ and 10^4 are on eather side of the transition region. In Fig 4.11 the distributions of the stress tensor components are plotted for $tau0 = 10^4$.

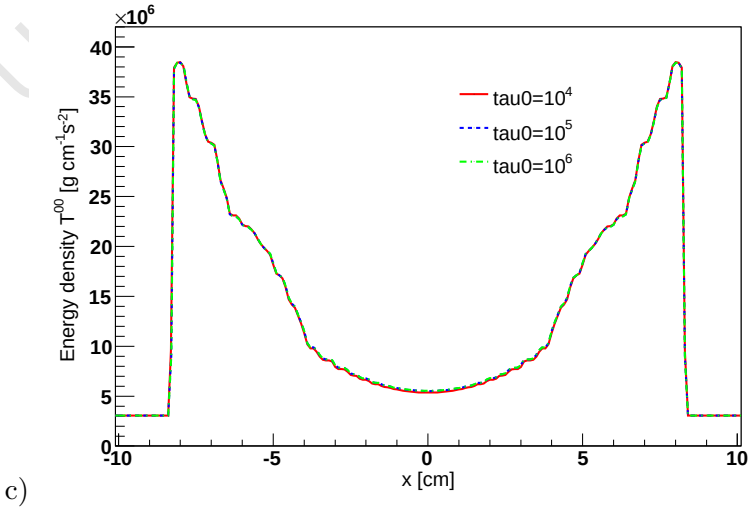
Setting $eta0 = 10^6$, **TNonideal_xy** obtains a solution for $tau0 \geq 10^0$ and, as with all the previous cases, a transition occurs as $tau0$ is varied. The energy density cross-sections for $tau0 = 10^0$, 10^1 and 10^2 are plotted versus that of the corresponding ideal system, Fig 4.12. While Fig 4.13 shows the stress tensor components for $tau0 = 10^1$.



a)



b)



c)

Figure 4.10 – The cross-section of the energy density (T^{00}) distribution after 180 time steps (5.4×10^{-5}s) for the viscosity multiplicative factor $eta0 = 10^4$ and various values of the relaxation time's multiplicative factor $tau0$. In (a) the solution to the associated ideal hydrodynamic system is also plotted.

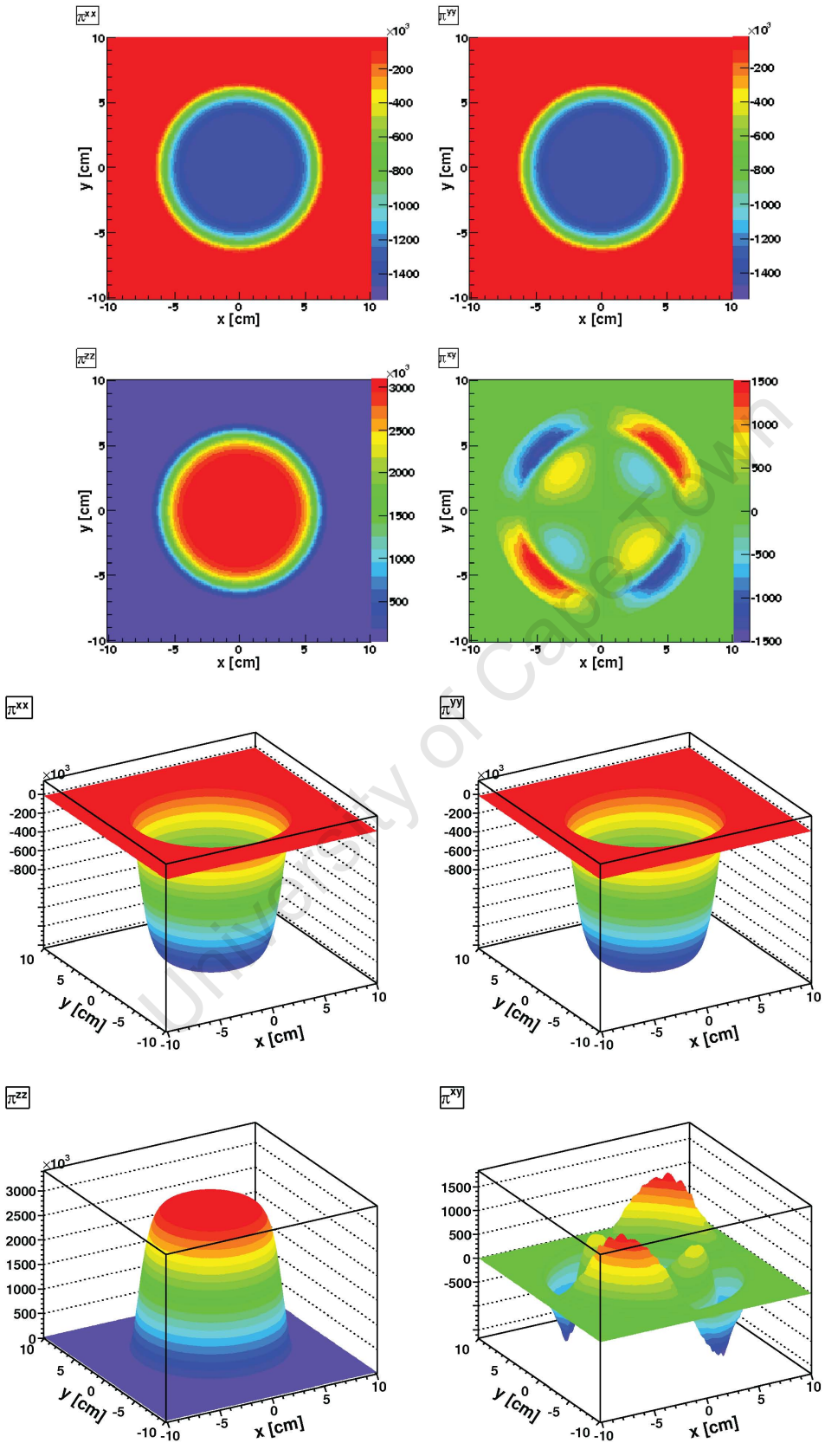
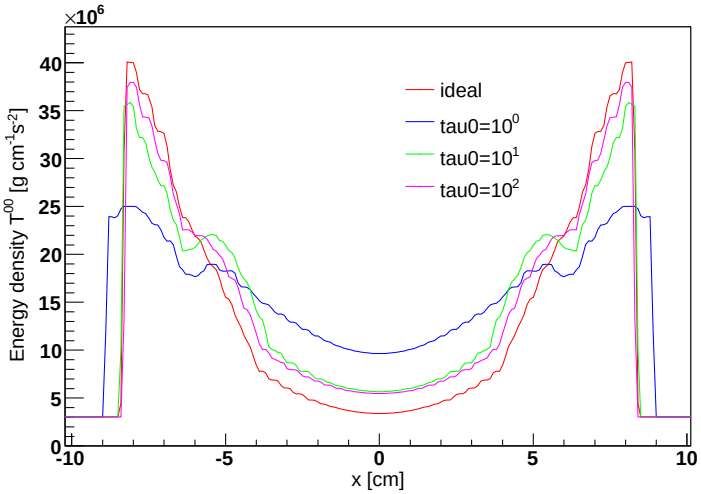


Figure 4.11 – The stress tensor components π^{ij} [g cm^{-1}s^{-2}] for $eta0 = 10^4$ and $tau0 = 10^4$, after 180 time steps (5.4×10^{-5}s)



a)

Figure 4.12 – The cross-section of the energy density (T^{00}) distribution after 180 time steps (5.4×10^{-5}s) for the viscosity multiplicative factor $eta0 = 10^6$ and various values of the relaxation time's multiplicative factor $tau0$. In (a) the solution to an ideal hydrodynamic system is also plotted.

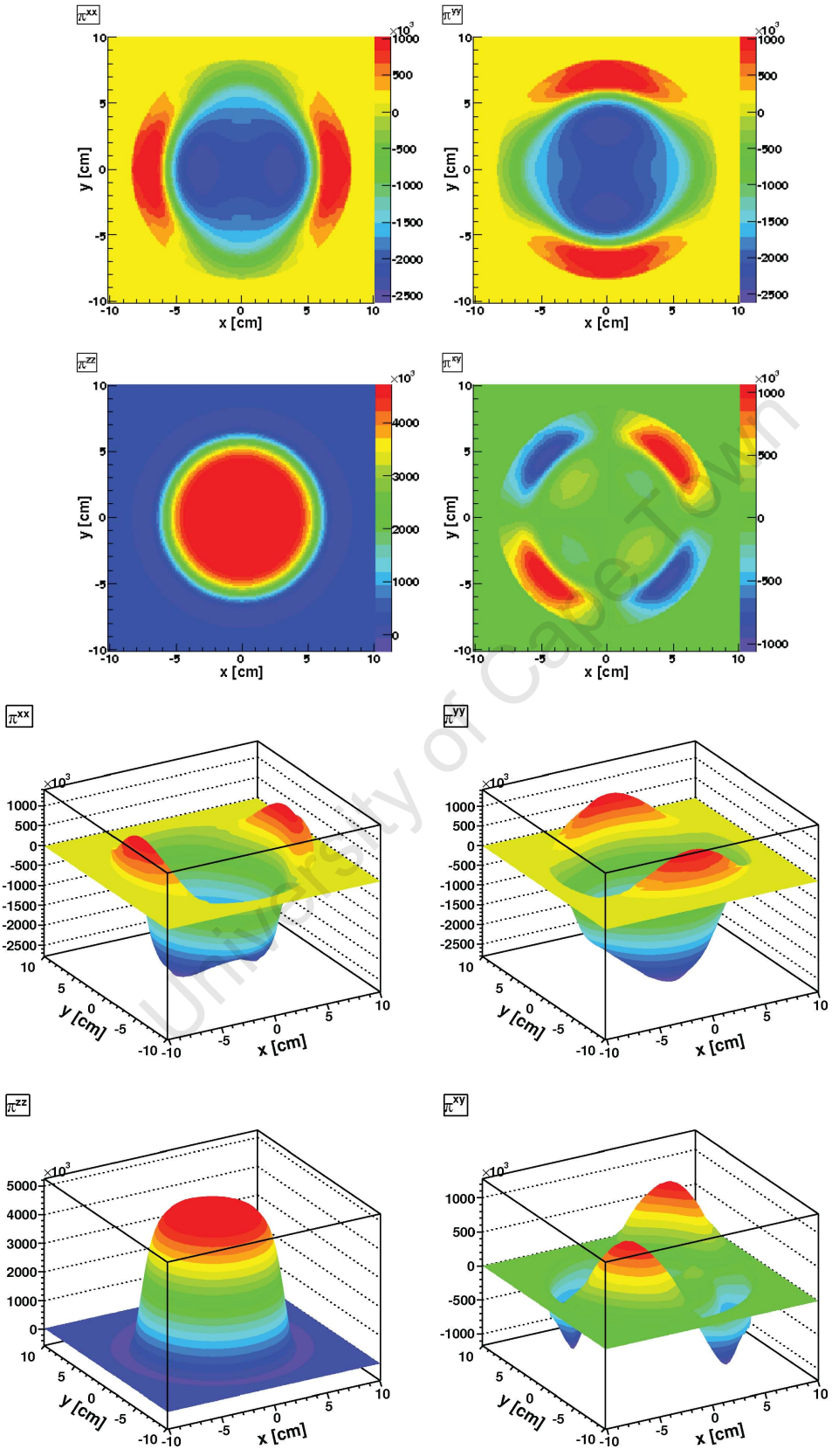


Figure 4.13 – The stress tensor components π^{ij} [g cm^{-1}s^{-2}] for $eta0 = 10^6$ and $tau0 = 10^1$, after 180 time steps (5.4×10^{-5}s).

Eccentricity

In this section we study the effect of viscosity, η, and the relaxation time τ_π on the eccentricity of the system. Changes in the eccentricity can be used to infer what changes one might expect in the elliptic flow. To recap, in heavy ion collisions, the eccentricity

$$\epsilon = \frac{\ll y^2 - x^2 \gg}{\ll y^2 + x^2 \gg} \tag{4.33}$$

where the angled brackets represent the energy weighted averages. As the x axis is co-linear with the impact parameter $\vec{b}$ the initial distributions are elongated in the y direction and the initial eccentricity is positive, for non-central collisions in heavy ion collisions.

The effects on the eccentricity is analysed for two separate sets of systems. In both systems, the geometry and system parameters are set to the same values as those used in Case A, above, except for *courant*. The parameter *courant* (= 0.45) is positive which means that the stepsize, dt, is changed thoughout the evolution such that the maximum cournat number ~ 0.45, at each timestep. With this setting, different systems evolve at different rates and the outputs do not occur at the same times. In order to calculate the difference in eccentricity for two different systems $(\epsilon_1 - \epsilon_2)$ at a specific time, the time dependence of the eccentricity is interpolated using a cubic spline interpolation. Again, the values of η and τ_π are altered through the multiplicative parameters $eta0$ and $tau0$.

For the first set of systems the initial conditions give rise to a shock wave during the expansion while in the second they do not. The first initial condition consists of static elliptic Woods-Saxon distributions with an extension of 2 cm in the y direction and 1 cm in the x direction and a skin width $a = 10\,\mathrm{d}x$ in both directions [Initialise(0,0,0,0,1); $xsi\_y = 2$; $xsi\_x = 1$; $a = 10$]. These are for the energy density T^{00} and the mass density of the gas $Nbnr$. Since it is static, the initial x and y components of the velocity are zero as are the momenta in these two directions; T^{0x} and T^{0y}. The initial pressure is obtained using the equation of state [$Gamma = 1.\dot{6}$] and the energy density. The heat flux components and the bulk viscosity are initialised to zero. The components of the stress tensor are initialised as $\pi_{xx} = -0.1 * \frac{P}{6}$, $\pi_{yy} = -0.1 * \frac{P}{6}$, $\pi_{zz} = 0.1 * \frac{P}{3}$ and $\pi_{xy} = 0$ [$fred = 0.1$].

For the system with a shock wave, the initial distribution for the energy density has a maximum and minimum of 1.52×10^8 and 1.52×10^6 $\mathrm{g\,cm^2 s^{-2}}$, respectively [$e0$ =1.52e8; $e4$ =1.52e6]. The density has a maximum and minimum of 1.292×10^{-2} and 1.292×10^{-3} $\mathrm{g\,cm^{-3}}$, respectively [$b0$ = 1.292e-2, $b4$ = 1.292e-3]. This is equivalent to a region of gas which has a maximum pressure 100 times that of standard pressure and a temperature 10 times standard temperature which expands into gas at STP. In the system without a shock wave the initial distributions have the same maximum values, while the minimum values are zero [$e0 = b0 = 0$]. This corresponds to the region of gas expanding into a vacuum.

To calculate the eccentricity in the situation where the "background" energy density is non-zero, the difference between the energy density and that of the "background" is used for the weighted average in the calculation of the eccentricity. We are interested in the disturbances above the background as the systems evolve. In the analysis which

follows the time dependence of the eccentricity of nonideal systems are compared to that of the corresponding ideal system. With the initial conditions described above, all systems start with an initial eccentricity $\epsilon = 0.6$.

For the first analysis the parameter $eta0 = 1$ and $tau0$ is varied. This leaves η unchanged while τ_π is increased by the factor $tau0$. It is found that with $eta0 = 1$ a solution is only obtained when $tau0 \geq 10^6$ for the systems with a shock wave and $tau0 \geq 10^5$ for systems without a shock wave. It is at these values that τ_π is sufficiently large for the values of dt used during the simulation. In the case of the systems with a shock wave, the eccentricity of the ideal system is plotted against time in Fig 4.14 (a), while (b) shows the absolute difference between the eccentricity of the nonideal systems the ideal system for $tau0 = 10^6$, 10^7 and 10^8. Fig 4.14 (c) and (d) show the similar information, but for the systems without a shock wave. Also, the absolute difference in eccentricities in this case is shown for $tau0 = 10^5$, 10^6 and 10^7. The eccentricities of the ideal and nonideal systems are not plotted on the same set of axes as they are too close to be easily distinguishable. The eccentricity of the nonideal systems are very close to

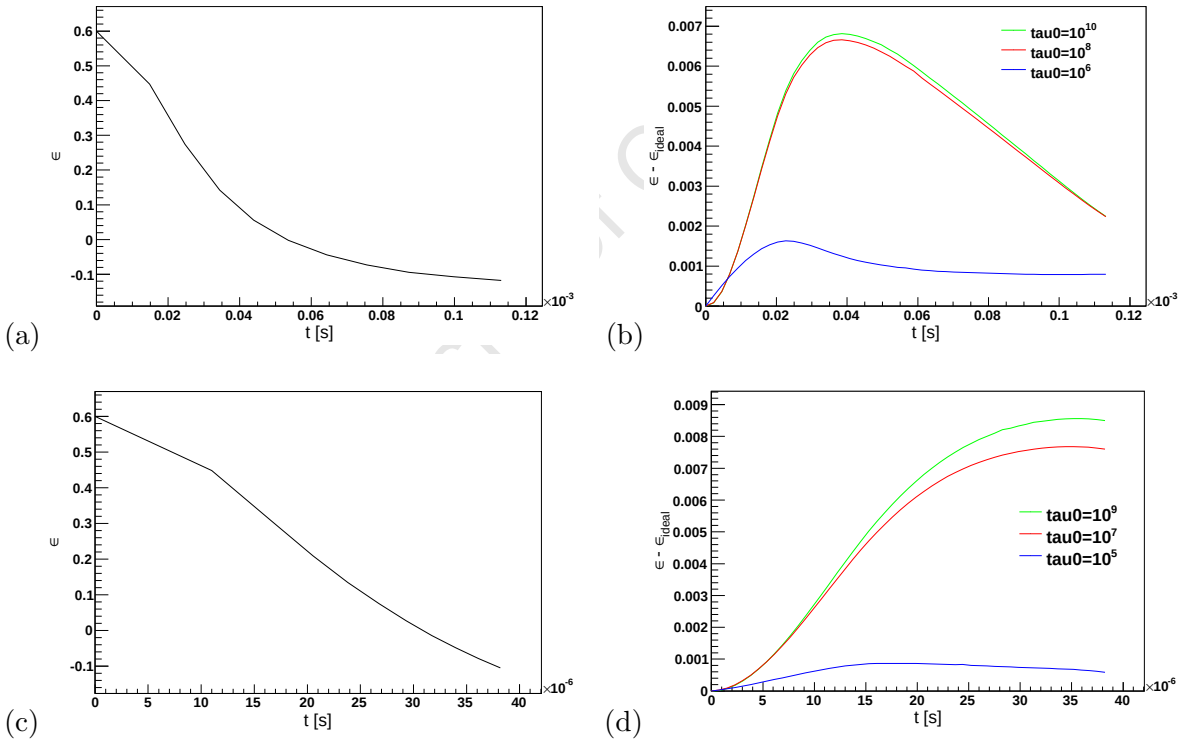


Figure 4.14 – The time dependence of the eccentricity of the ideal system with a shock wave is plotted in (a). Graph (b) shows the absolute difference between the eccentricities of the nonideal systems, for various values of $tau0$ and $eta0 = 1$, and that of the ideal system. Graphs (c) and (d) display the same, but for systems without a shock wave

that of the ideal one. Also, the systems with a shock wave have a more rapid change in eccentricity compared with those without the shock wave, as one might have expected.

For the second analysis, $tau0 = 1$ and $eta0$ is varied. Recall that this has the effect of increasing the value of η and τ_π by a factor of $eta0$. For the systems with a shock wave solutions were only found when $eta0 = 10^6$, while for those without a shock wave solutions were found when $10^5 \leq eta0 \leq 10^9$ [values above 10^9 were not tested]. In Fig 4.15 (a) the eccentricity of the different nonideal systems without a shock wave are plotted along with that of the corresponding ideal system. In (b) the absolute difference between the eccentricities of the nonideal systems and that of the ideal system $(\epsilon - \epsilon_{ideal})$ is plotted. As the viscosity, η, is increased the eccentricity of the system decreases ever

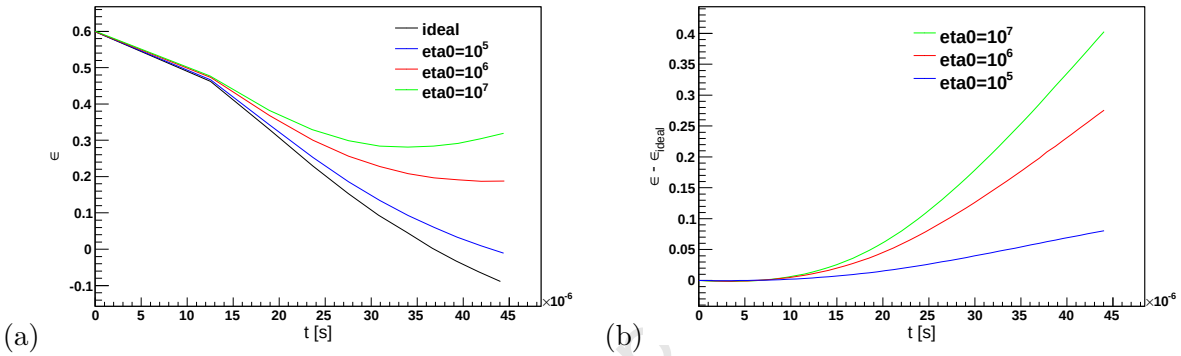


Figure 4.15 – The time dependence of the eccentricities of the ideal and nonideal systems with a shock wave are shown in (a). The nonideal systems plots are for $tau0 = 1$ and different values of $eta0$. The absolute difference between the eccentricities of the nonideal systems and that of the ideal system are shown in (b).

slower. If fact for $eta0 = 10^7$ the viscosity is so large that the eccentricity begins to increase after about 3.5×10^{-5} s. This is related to the phenomenon realised in Case B above, where the system has such a high viscosity that it begins to contract after the initial expansion. For $eta0 = 10^6$ it seems as though the eccentricity is just about to start increasing at the point where the simulation ends. It would be interesting to analyse the effect of changing τ_π for the cases where $eta0 = 10^5$ and 10^6 and we do so in the following.

For the next analysis, the parameter $eta0 = 10^5$ and $tau0$ is varied. This has the effect of increasing η buy the factor $eta0$ and τ_π by the factor $eta0 * tau0$. Solutions were found when $tau0 \geq 10^1$ for systems with a shock wave and $tau0 \geq 10^0$ for those without a shock wave. In Fig 4.16 (a) the eccentricities of the nonideal systems with a shock wave, for $tau0 = 10^1$, 10^2 and 10^3, are shown as well as the eccentricity of the corresponding ideal system. In (b) the absolute difference between the eccentricities of the nonideal systems and that of the ideal system is shown. The similar data is show in Fig 4.16 (c) and (d), but for systems without a shock wave. In this case the graphs are for $tau0 = 10^0$, 10^1 and 10^2.

For the final analysis, $eta0 = 10^6$ and, again, $tau0$ is varied. For both systems, with and without a shock wave, solutions were found when $tau0 \geq 10^0$. The eccentricities when $tau0 = 10^0$, 10^1 and 10^2 are shown in Fig 4.17 (a) and (c) for systems with and without a shock wave, respectively. In both cases, they are plotted with the eccentricity of the corresponding ideal systems. In (b) and (d) the absolute difference between the

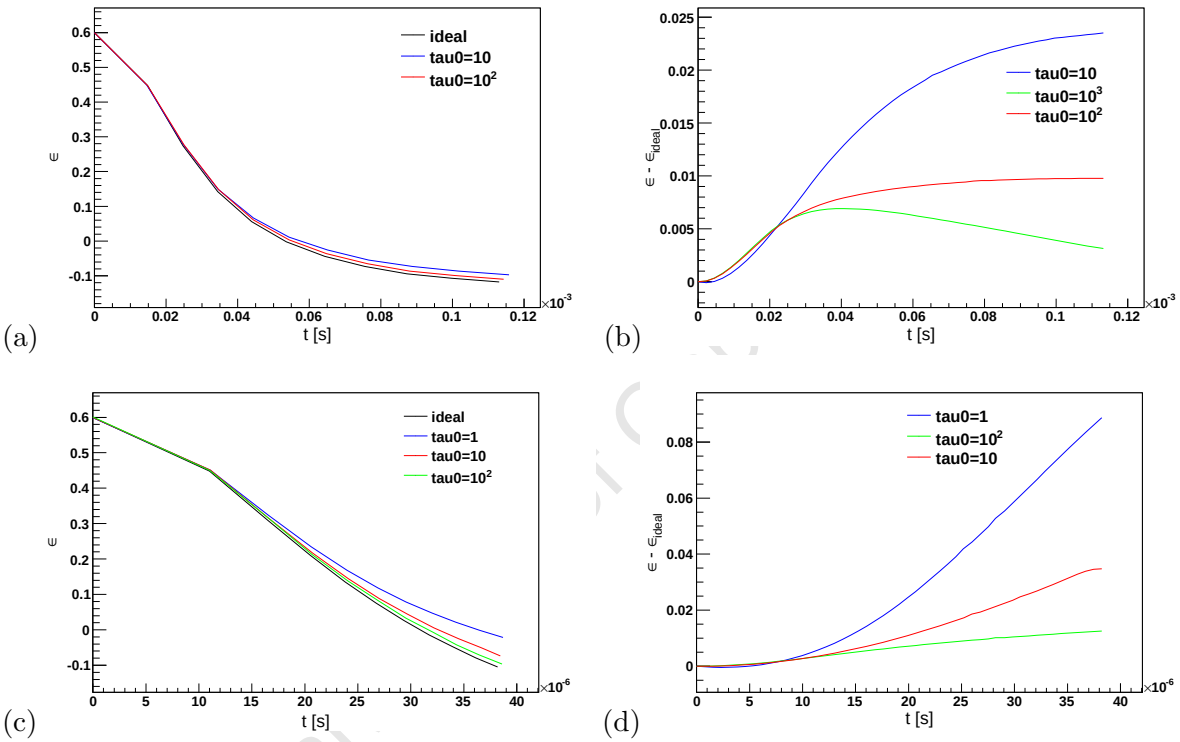


Figure 4.16 – The time dependence of the eccentricities of the nonideal systems with a shock wave and the corresponding ideal system are show in (a). The same is shown in (c) for the systems without a shock wave. The absolute difference between the eccentricities of the nonideal and ideal systems are shown in (b) and (d) for systems with and without a shock wave, respectively. In all cases $eta0 = 10^5$ and $tau0$ is varied for the nonideal systems.

eccentricities of the nonideal systems and the ideal system are shown for those systems with and without the shock wave, respectively

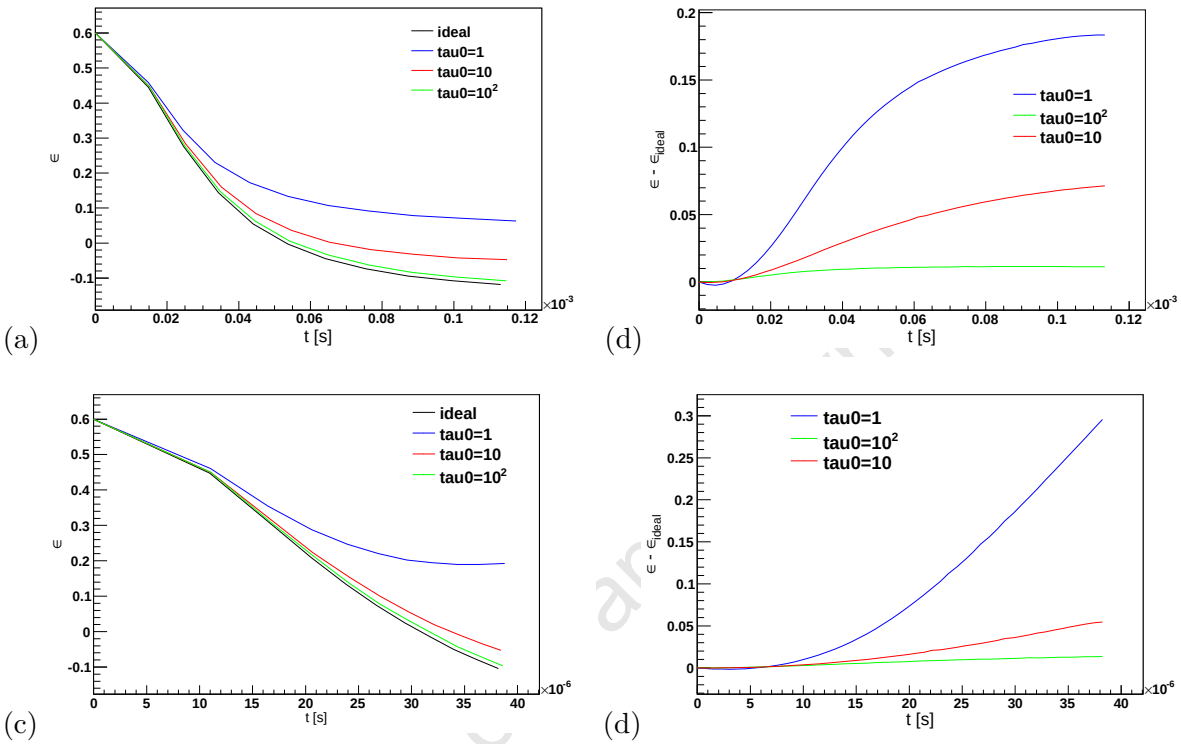


Figure 4.17 – The time dependence of the eccentricities of the nonideal systems with a shock wave and the corresponding ideal system are show in (a). The same is shown in (c) for the systems without a shock wave. The absolute difference between the eccentricities of the nonideal and ideal systems are shown in (b) and (d) for systems with and without a shock wave, respectively. In all cases $eta0 = 10^6$ and $tau0$ is varied for the nonideal systems..

The evolution of the eccentricity becomes more like that of the ideal systems as the relaxation time τ_π is increased. So although the actual distributions diverge from the ideal system's distributions as τ_π is increased, the eccentricity evolution diverges more as τ_π decreases.

4.3.2 Ultra-Relativistic Values

For the non-relativistic gas, section 4.3.1, it was found that the relaxation time τ_π suggested a much smaller value for dt than the courant number suggests. Thus a much larger number of time steps would be required to preform a simulation for a non-ideal versus and ideal system. Fortunately in relativistic heavy ion collision systems this is not the case. For example if we consider a gas of quarks and gluons near the freezeout curve then the temperature is $T = 170$ MeV and the energy density is $\varepsilon = 1$ GeV fm^{-3}. With the parton scattering cross section, $\sigma \sim 8$ mb, we can approximate the shear viscosity

for the ultra-relativistic system [116] as

$$\eta = \frac{3}{5\pi}\frac{T}{\sigma} = 41 \text{ MeV fm}^{-2}\text{c}^{-1}. \tag{4.34}$$

Now using the equation of state, $P = \frac{\varepsilon}{3}$, for a gas of massless particles we have from [83]

$$\tau_\pi = \frac{3}{2}\frac{\eta}{P} = 0.19 \text{ fm c}^{-1}. \tag{4.35}$$

which would suggest $\mathrm{d}t \sim 0.01$ fm c^{-1}. Alternatively the **FCTHydro** routines require a courant number, $\frac{v\,\mathrm{d}t}{\mathrm{d}x} \leq \frac{1}{2}$. Since the upper bound for the velocity is the speed of light, we can obtain an approximate bound $\mathrm{d}t$. If the size of the initial distributions are of the order of the diameter of a Pb nucleus a reasonable spatial grid spacing would be $\mathrm{d}x \sim 0.1$ fm and as such $\mathrm{d}t \sim 0.05$ fm c^{-1}. This is comparable with the value suggested by τ_π.

Chapter 5

Analytic Landau Hydrodynamics and Cooper-Frye Freeze-out

It was shown in section §3.3.2 that a massive simulation is required to produce the freeze-out curve in (1+1)D hydrodynamics with Landau initial conditions, in the case where the freeze-out to initial temperate ratio, $\frac{T_{fo}}{T_0}$, is small. In fact it was shown that to produce the curve in the instance where $\frac{T_{fo}}{T_0} = 0.1$ would require a simulation grid with 200,000 cells and 400,000 timestep iterations, which is not feasible. Thus a method other than that described in section §3.3.2 must be used for small $\frac{T_{fo}}{T_0}$.

It is infact possible to calculate the freeze-out spectrum from Landau hydrodynamics analytically. Here we illustrate this in the case of a gas of massless particles which obey Boltzmann statistics. The spectrum of particles emitted from the freeze-out curve, as determined from equations (3.26) - (3.28), is

$$\frac{\mathrm{d}N}{\mathrm{d}y} = 2\pi \int_\sigma \frac{2T_{fo}^3}{\cosh^3(y-\alpha)} \left[\frac{\partial t}{\partial \xi} \sinh y - \frac{\partial x}{\partial \xi} \cosh y \right] \mathrm{d}\xi, \tag{5.1}$$

where y is the particle rapidity, $\alpha = \arctan(v)$ is the flow rapidity and

$$t = t(\xi), \qquad x = x(\xi)$$

are parametric equations for the temporal and spatial coordinates of the freeze-out curve, σ. All that is needed is the appropriate parametric parameters for the freeze-out curve. In the following, the spacelike and timelike sections of the freeze-out curve are dealt with separately.

5.1 Spacelike curve

The spacelike section of the freeze-out curve is the section which occurs in the region described by simple Riemann waves (section §3.3.2). In this region the freeze-out curve can be parametrised by t, with the positive orientation of the curve define by increasing

t. Thus the freeze-out curve $\sigma^\nu = [t, x(t), 0, 0]$, where

$$x(t) = \frac{v_{max} - c_s}{1 - v_{max} c_s} t, \tag{5.2}$$

with

$$\alpha_{max} \equiv \operatorname{arctanh}(v_{max}) = -\frac{1}{c_s} \ln\left(\frac{T_{fo}}{T_0}\right), \tag{5.3}$$

and c_s the speed of sound. Thus for the spacelike section equation (5.1) becomes

$$\frac{\mathrm{d}N}{\mathrm{d}y} = \frac{4\pi T_0^3 t_{max}}{\cosh^3(y - \alpha_{max})} \left[\sinh y - \frac{v_{max} - c_s}{1 - v_{max} c_s} \cosh y\right],$$

where y is rapidity. By introducing the speed of sound rapidity, $\alpha_s = \tanh(c_s)$, the equation for the spectrum of particles frozen out from the spacelike section reduces to the simpler form

$$\frac{\mathrm{d}N}{\mathrm{d}y} = \frac{4\pi T_0^3 t_{max}}{\cosh^3(y - \alpha_{max})} \left[\frac{\sinh(y + \alpha_s - \alpha_{max})}{\cosh(\alpha_{max} - \alpha_s)}\right]. \tag{5.4}$$

5.2 Timelike curve

The timelike section occurs in the region were Landau's solution is valid. Here we can parametrise the curve using the flow rapidity, $\alpha = \operatorname{arctanh} v$, where α takes values from α_{max} to 0 for the curve to be positively orientated. This requires knowledge of the partial derivatives $\frac{\partial x}{\partial \alpha}$ and $\frac{\partial t}{\partial \alpha}$, which are calculated from equations (3.24). In terms of the potential function, from equation (3.23);

$$\chi(w, \alpha) = -\frac{T_o l}{c_s} e^w \int_{c_s \alpha}^{-w} e^{w'(\beta+1)} \mathrm{I}_0(\beta \sqrt{w'^2 - c_s^2 \alpha^2}) \mathrm{d}w', \tag{5.5}$$

the partial derivatives are

$$\frac{\partial x}{\partial \alpha} = \frac{e^{-w}}{T_0} \left[\left(\frac{\partial^2 \chi}{\partial w \partial \alpha} - \frac{\partial \chi}{\partial \alpha}\right) \sinh \alpha - \left(\frac{\partial \chi}{\partial \alpha} - \frac{\partial^2 \chi}{\partial \alpha^2}\right) \cosh \alpha\right], \tag{5.6}$$

$$\frac{\partial t}{\partial \alpha} = \frac{e^{-w}}{T_0} \left[\left(\frac{\partial^2 \chi}{\partial w \partial \alpha} - \frac{\partial \chi}{\partial \alpha}\right) \cosh \alpha - \left(\frac{\partial \chi}{\partial \alpha} - \frac{\partial^2 \chi}{\partial \alpha^2}\right) \sinh \alpha\right].$$

The partial derivatives can then be substituted into

$$\frac{\mathrm{d}N}{\mathrm{d}y} = 2\pi \int_{\alpha_{max}}^{0} \frac{T_0^3}{\cosh^3(y - \alpha)} \left[\frac{\partial t}{\partial \alpha} \sinh y - \frac{\partial x}{\partial \alpha} \cosh y\right] \mathrm{d}\alpha \tag{5.7}$$

and the spectrum found by performing the integration.

We are now able to calculate the $\frac{\mathrm{d}N}{\mathrm{d}y}$ spectra for Landau hydrodynamics with Cooper-Frye freeze-out. The full $\frac{dN}{dy}$ spectrum is obtained using the symmetry of the

system about $y = 0$.

In Fig 5.1 we present the $\frac{\mathrm{d}N}{\mathrm{d}y}$ spectra for freeze-out temperatures $\frac{T_{fo}}{T_0} = 0.2$, 0.4, 0.6 and 0.8, where Gaussian quadrature with 20 nodes was used to evaluate the integral in equation (5.7). These can be compared to the results in Fig. 7 from [108], which were

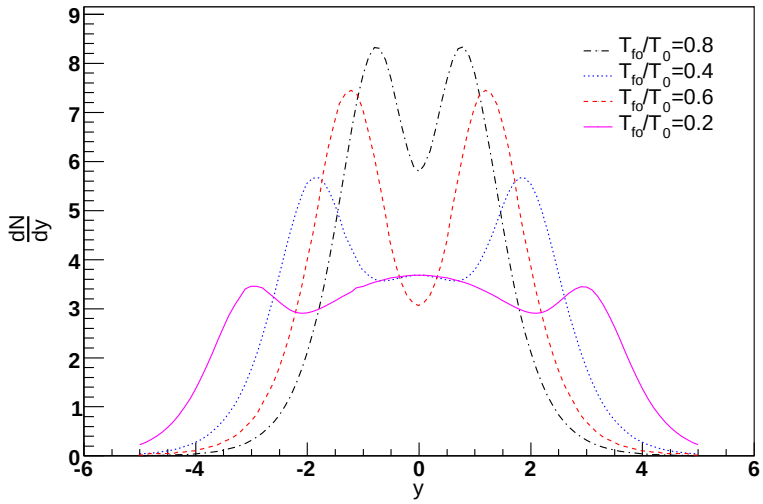


Figure 5.1 – The freeze-out spectra from Landau hydrodynamics and Cooper-Frye freeze-out for $l = 1$, $c_s = \frac{1}{\sqrt{3}}$ and various values of $\frac{T_{fo}}{T_0}$.

obtained by performing a numerical simulation of the system hydrodynamic system. As identified in [108], the spectra are far from the Gaussian distribution generally associated with Landau hydrodynamics. This due to the contribution of the spacelike section of the freeze-out curve. Fig 5.2 shows this more clearly, where the contribution due to the spacelike and timelike sections of the freeze-out curve are plotted separately. The contribution from the timelike section has a Gaussian like distribution, while that from the spacelike is a double peaked spectrum and is not negligible. One could then conclude that the "true" solution from Landau hydrodynamics is not valid, as a two peak spectrum is not seen in the $\frac{\mathrm{d}N}{\mathrm{d}y}$ spectra from heavy ion collisions. However, as we will show later, the contribution of the timelike section becomes negligible when the collision energy is high enough.

It is interesting to note the negative values from the spacelike section at mid-rapidity. These are due to an anomaly of Cooper-Frye freeze-out, where one can obtain negative contributions from particles which are travelling "back" into the fluid from the freeze-out hypersurface. This anomaly was first addressed in [71] where a Heaviside step function $\theta(p_\mu u^\mu)$ is used to prevent negative contributions. However, this correction unfortunately violates energy-momentum conservation.

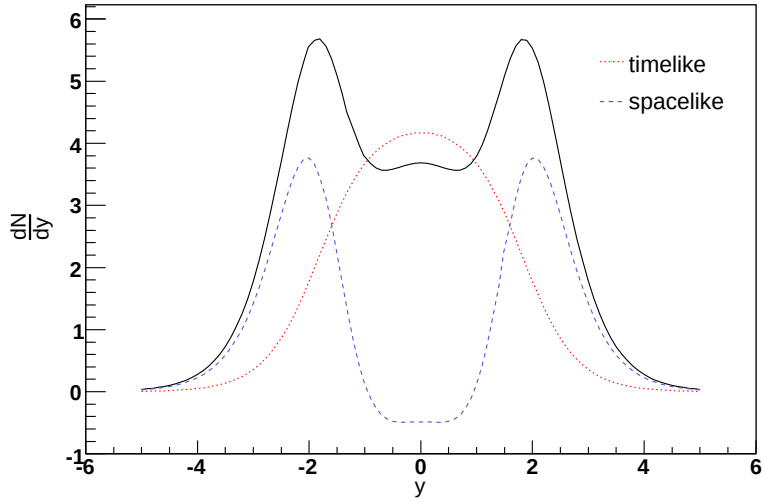


Figure 5.2 – The contribution to the freeze-out spectra from the spacelike and timelike sections of the freeze-out curve for $l = 1$, $c_s = \frac{1}{\sqrt{3}}$ and $\frac{T_{fo}}{T_0} = 0.4$.

5.3 Test Case for Analytic Freeze-out

Let us consider a central, $\sqrt{s}_{NN} = 200$GeV Au-Au collision and apply Landau hydrodynamics. Firstly we need to determine the initial temperature and longitudinal extension. We will assume that the nuclei are Lorentz contracted into discs and so need the gamma factor for a $\sqrt{s_{NN}} = 200$GeV collision. In the centre of mass the two nuclei have four momenta $P_1 = [E, \vec{p}]$ and $P_2 = [E, -\vec{p}]$. Thus, using $P_i^2 = m_N^2$, $E = \gamma m_N$ and $\vec{p} = \gamma \vec{v} m_N$,

$$\begin{aligned} s_{NN} &= (P_1 + P_2)^2 = P_1^2 + P_2^2 + 2P_1 \cdot P_2 \\ &= 2m_N^2 + 2E^2 + 2\vec{p}^2 \\ &= 2m_N^2(1 + \gamma^2 + v^2\gamma^2) \\ &= 4m_N^2\gamma^2, \end{aligned}$$

which gives

$$\gamma = \frac{\sqrt{s_{NN}}}{2m_N}. \tag{5.8}$$

To calculate the volume of the cylinder we just need the radius which is given approximately by

$$r \sim 1.2A^{1/3}\,\text{fm}, \tag{5.9}$$

with A the nucleon number of the atoms. Thus the volume of the cylinder is

$$V = \pi r^2 \cdot \frac{2r}{\gamma} = \frac{2\pi r^3}{\gamma} = \frac{2\pi A(1.2)^3}{\gamma}. \tag{5.10}$$

Then the initial energy density for the collision

$$\varepsilon = \frac{E}{V} = \frac{\sqrt{s_{NN}}A\gamma}{2\pi A(1.2)^3}$$
$$= \frac{s_{NN}}{4(1.2)^3\pi m_N}$$

It is interesting to note that for this geometry, the energy density is independent of the size of the nucleus. Now that we have the energy density we can finally calculate the temperature, assuming that the equation of state is that of a massless Boltzmann gas,

$$T = \left[\frac{37\pi^2(0.197)^3\varepsilon}{30}\right]^{1/4}. \tag{5.11}$$

Applying this information to our example, the half extension of the system at $t = t_0$ is

$$l = \frac{r}{\gamma} = \frac{2 \cdot 1.2 \cdot (197)^{1/3} \cdot 0.938}{200} = 0.065495\,\text{fm}, \tag{5.12}$$

the initial energy density is

$$\varepsilon = \frac{(200)^2}{4\pi(1.2)^3 \cdot 0.938} = 1964\,\text{GeV/fm}^3, \tag{5.13}$$

and the initial temperature

$$T_0 = 3.68\,\text{GeV}. \tag{5.14}$$

These values are wildly different from those predicited by the Bjorken model; $\varepsilon \sim 3GeV$ and extension of initial distribution ~ 1 fm. This is because for the Landau system, unlike the Bjorken model, it has been assumed that the matter has stopped during the collision and that thermalisation has occured instataneously.

Substituting the values obtained above into the solutions for the hydrodynamics along with
$c_s = \frac{1}{\sqrt{3}}$ and $T_{fo} = 0.17$GeV, we obtain the spectrum in Fig 5.3. Although, for rapidities $|y| < 3$, the distribution is Gaussian like, it is much broader than Landau's Gaussian distribution [32]

$$\frac{\mathrm{d}N}{\mathrm{d}y} \sim e^{-\frac{y^2}{2L}}, \qquad L = ln(\gamma_{cm}) \tag{5.15}$$

which he obtained by making certain simplifying assumptions. Srivastava et al [117] avoided some of the simplifying assumptions and arrived at the distribution

$$\frac{\mathrm{d}N}{\mathrm{d}y} \sim I_0(p) + \beta|w_f|\frac{I_i(p)}{p}, \tag{5.16}$$

for small rapidities, where $p = \beta\sqrt{w_f^2 - c_s^2y^2}$ and $w_f = \ln\left(\frac{T_{fo}}{T_0}\right)$. Their spectrum is also much broader than Landau's Gaussian one, but not as broad as the spectrum obtained from combining full Landau hydrodynamics and Cooper-Frye freeze-out. In Fig 5.4 we

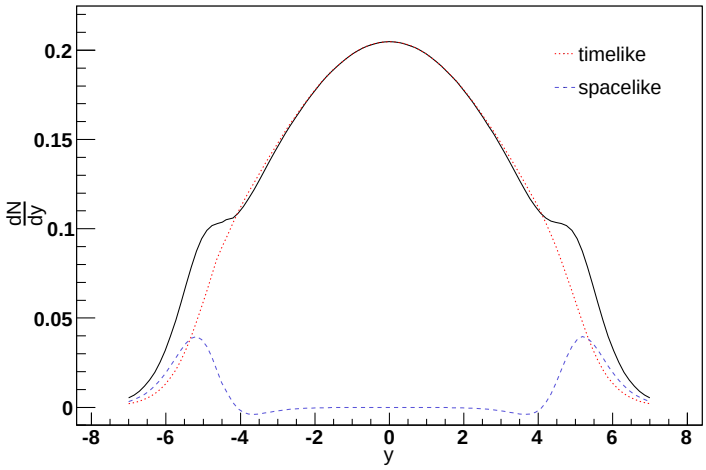
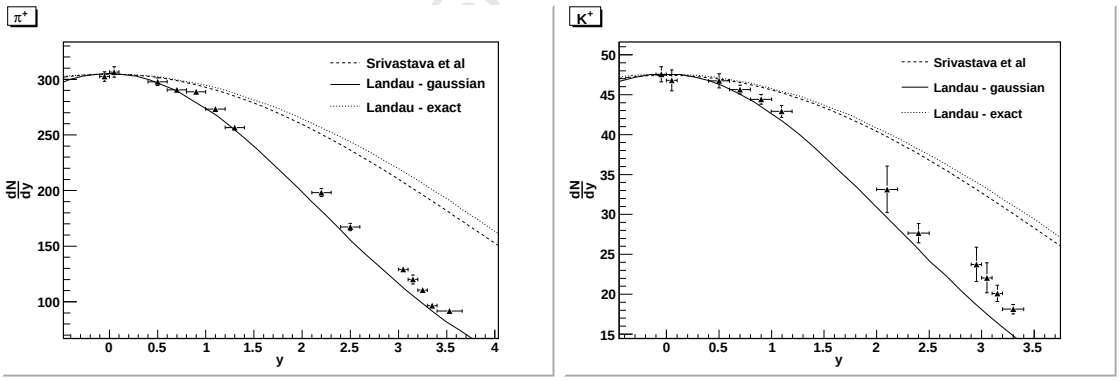


Figure 5.3 – $\frac{dN}{dy}$ for massless Boltzmann particles using Landau's hydrodynamics for a $\sqrt{s}_{NN} = 200$ AGeV, Au-Au collision. The figure shows the contribution from the spacelike (dashed line) and timelike (dotted line) sections of the freeze-out curve. The solid line indicates the sum of these two contributions.

have compared the rapidity spectra from BRAHMS [118] for π^+ (a) and K^+ (b) with Landau's Gaussian, Srivastava et al [117] and Landau hydrodynamics with Cooper-Frye freeze-out. The theoretical distributions were calculated using the values for the parameters found above and have been normalised such that their maxima are the same. Amazingly it is Landau's Gaussian distribution which describes the data most accurately.



(a) (b)

Figure 5.4 – Comparison of π^+ (a) and K^+ (b) spectra from BRAHMS [118] with Landau's Gaussian, Srivastava et al [117] solution and our full Landau hydrodynamics with Cooper-Frye freeze-out.

Srivastava et al pointed out that their distribution narrows towards the Gaussian distribution when $c_s \to 0$. This is also true when Landau hydrodynamics is combined with Cooper-Frye freeze-out, as is evident in Fig 5.5 where the same parameters were used

as above, but three different values of the speed of sound are chosen. The distributions were normalised such that their maxima are equal. However, as $c_s \to 0$ the time it takes

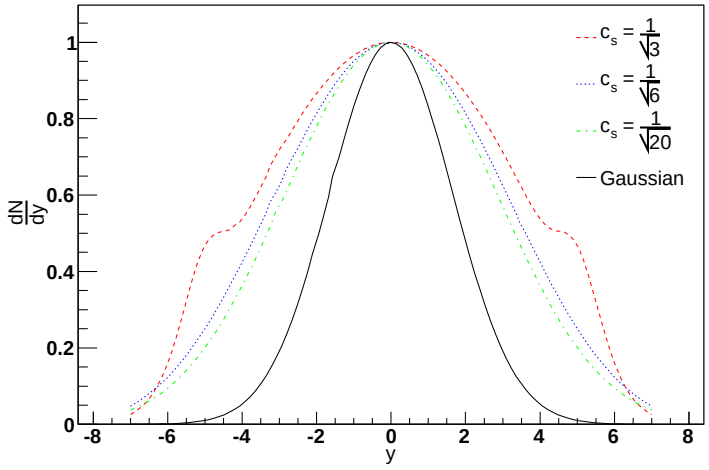


Figure 5.5 – The dependence of $\frac{\mathrm{d}N}{\mathrm{d}y}$ on the speed of sound for the exact solution using Landau hydrodynamics and Cooper-Frye freeze-out..

for the system to cool diverges, and we end up with a system which does not expand at all.

If one now assumes that during the collision only part of the energy is deposited in the collision zone then the initial temperate for the hydrodynamics is lowered and the ratio $\frac{T_{fo}}{T_0}$ increases. This will narrow the width of the distribution obtained from Landau's solution coupled with Couper-Frye, however the effect of the leading edges also increases and becomes a significant contribution. This results in the three peak spectrum which is defintely not repreduced in the profile of the actual data.

Chapter 6

AGS data and the E/N Freeze-out Criterion

This chapter is distinct from the rest of the thesis, in that it is not related to hydrodynamics. In this chapter a statistical-thermal model is used to fit the chemical freezeout points for preliminary[119] data from AGS. These fits are performed using the package THERMUS [101]. The data consist of 4π yields from central Au-Au collisions at nominal beam energy 2, 4, 6 and 8 AGeV. The fits are then compared to the Cleymans-Redlich freeze-out criterion.

6.1 Theory

Statistical-thermal models are able to characterise the production of a large number of hadrons using very few parameters. They have been applied to high- energy collisions from as early as the 50's and 60's, with the work of Fermi [120], Landau [121] and Hagedorn [122]. Since then, statistical-thermal models have been very successful in describing data for a wide range of energies in both heavy-ion and elementary collisions [123, 124, 125, 126, 127, 128, 129, 130].

In order for statistical-thermal models to be valid, the system must be in chemical equilibrium. This equilibrium could possibly be obtained through the filling of hadronic phase space at hadronisation or it could evolve from an initial non-equilibrium state. For the first process, the phase space is filled via the principle of maximising entropy and requires no scattering. In the second, the evolution from a state of non-equilibrium occurs via re-scattering. Although it was shown [131] that two particle scatterings are insufficient to equilibrate the system, multi-particle collisions are very effective near the phase change transition region.

In high-energy collisions the baryon (B), charge (Q) and strangeness (S) quantum numbers are conserved charges. When using the statistical-thermal model, there are typically two different ensemble choices for treating each of these quantum numbers. In the grand-canonical ensemble, chemical potentials are introduced to allow for perturbations about conserved averages and is, understandably, only feasible if the conserved number is large. However, in the canonical ensemble the charges are conserved exactly. This ensemble is important in smaller and colder systems, for example, elementary collisions.

Not all the conserved charges need be treated using the same ensemble. It is possible to have a mixed-canonical ensemble [132], where, for example, B and Q are treated grand-canonically while S is treated canonically. This particular mixed ensemble is generally used at lower energy heavy-ion collisions, such as those at SIS. Although the temperatures are low, the large number of nucleons in the collision particles ensures that B and Q are sufficiently large to use the grand-canonical ensemble, while S is zero.

It has been suggested that, when using canonical systems, there should be two volume parameters [133, 134, 135]. The one is the freeze-out volume V_f and the other a correlation volume V_c. The freeze-out volume determines the overall normalisation of particle yields, while V_c is the volume over which local conservation of the quantum numbers must occur. By choosing $V_c < V_f$ the inherent strangeness suppression of the mixed-canonical ensemble is boosted. Another way to account for possible strangeness suppression, due to deviation from complete strangeness chemical equilibrium, is the introduction of the phenomenological factor γ_S [102, 103]. For each of the particle species, their Boltzmann factor is multiplied by the factor $\gamma_S^{|S_i|}$, where $|S_i|$ is the number of valence strange quarks in the hadron. So, strangeness suppression can be enhanced in the mixed-canonical ensemble by choosing either $V_c < V_f$ or $\gamma_s < 1$.

From the fits to various AA collision systems, a universal chemical freeze-out criterion, that the energy per hadron is 1 GeV at chemical freeze-out, was proposed [125, 126]. Since its introduction, this criterion has proved very accurate in describing statistical-thermal model fits. Two other freeze-out criteria which have been proposed are that the total baryon density is fixed [136] and the ratio of entropy density to the temperature cubed $\frac{s}{T^3}$ is fixed [137, 138].

Mixed-Canonical Ensemble

Due to the energies of the AGS systems fitted in this chapter, the mixed-canonical ensemble is chosen. For this ensemble and Boltzmann statistics, the partition function is given as

$$Z_S = \frac{1}{2\pi}\int_{-\pi}^{\pi} \mathrm{d}\phi_S e^{-iS\phi_S} \exp\left\{ \sum_{\text{species } i} \frac{g_i V}{(2\pi)^3} \int \mathrm{d}^3p \; e^{-(E_i-\mu_i)/T + iS_i\phi_S} \right\} \tag{6.1}$$

in terms of the volume, V and temperature, T, for particles with energy E_i, degeneracy g_i and chemical potential

$$\mu_i = B_i\mu_B + Q_i\mu_Q, \tag{6.2}$$

and B_i, Q_i and S_i are the particle specific baryon, charge and strangeness quantum numbers. The particle number, entropy, pressure and energy then are obtained via,

$$\begin{aligned} N_i^S &= \frac{\partial \ln Z_S(\lambda_i)}{\partial \lambda_i}, \\ S^S &= \frac{\partial T \ln Z_S}{\partial T}, \\ P^S &= \frac{\partial T \ln Z_S}{\partial V}, \\ E^S &= -T \ln Z_S + TS^S + \sum_{\text{species } i} \mu_i N_i^S \end{aligned} \tag{6.3}$$

where the fictitious fugacity factor λ_i is introduced as a multiplicative factor in the species specific partition function.

6.1.1 The Fit

The data were taken by the E895 experiment for Au-Au runs at nominal beam energies 2, 4, 6 and 8 AGeV. After correcting for energy loss it was found that the actual beam energy of the 2 and 4 AGeV beams were in fact 1.85 and 3.91 AGeV, respectively [139]. No corrections were needed for the 6 and 8 AGeV runs. All 4π data were obtained from Gaussian fits to the rapidity spectra, as they are generally well described by Gaussians. The π^+, π^- and proton yields are obtained from [139, 140]. The yields for K^+, K^-, deuteron ^{2}H, triton ^{3}H and hellion ^{3}He are preliminary [119]. The 4π yields for these particles are presented in Table 6.1.

E_b	π^+[139]	π^-[139]	K^+	K^-	p[140]	^{2}H	^{3}He
8	96.9±1.1	130.7±0.4	19.43±1.94	2.44±0.24	153.1±0.3	30.6±0.3	0.97±0.01
6	75.7±1.1	104.0±0.2	12.1±1.2	1.16±0.12	154.9±0.4	34.7±0.2	1.47±0.01
4	46.3±0.8	76.0±0.2	5.65±0.57	0.37±0.04	139.7±0.3	34.9±0.2	2.53±0.07
2	19.2±1.3	36.1±0.3	0.78±0.07	0.00±0.07	113.7±0.2	37.0±0.1	6.10±0.05

Table 6.1 – 4π data from AGS for central Au-Au collisions with nominal beam energies 2, 4, 6 and 8 AGeV. The results for K^+, K^-, p, ^{2}H, ^{3}H and ^{3}He are preliminary [119].

As mentioned previously, the mixed-canonical ensemble is chosen for the statistical-thermal model fits. The correlation volume is assumed to be equal to the freeze-out volume and the charge chemical potential μ_Q is constrained by setting the ratio $B/2Q = 1.24662$. This is the initial ratio for Au-Au collisions. Thus the parameters fitted are the temperature T, baryon chemical potential μ_B, the fireball radius R and γ_S. For 6 and 8 AGeV, the fits were obtained using the pions, kaons, proton and hellion yields. For 4 AGeV, a fit was obtained using the pions, K^+, proton and hellion yields. While for 2 AGeV, a fit was obtained when the pions, kaons, proton and deuteron yields were used.

The fit values, obtained from THERMUS, are presented in Table 6.2 and are plotted in Fig 6.1. This figure also indicates the line corresponding to the $E/N = 1$ GeV freeze-out criterion. When the hellion and deuteron yields are ignored we are unable to obtain

Energy AGeV	T MeV	μ_B MeV	γ_S	R fm	χ^2
8	116.6±0.6	588.8±0.9	0.716±0.057	8.79±0.07	22.6
6	106.9±0.4	633.9±0.9	0.589±0.048	9.21± 0.05	6.35
4	94.82±0.62	695.1±1.5	0.719±0.074	9.08±0.09	2.92
2	77.34±0.36	752.6±0.6	0.302±0.025	9.82±0.05	0.256

Table 6.2 – THERMUS fit to 4π data in Table 6.1.

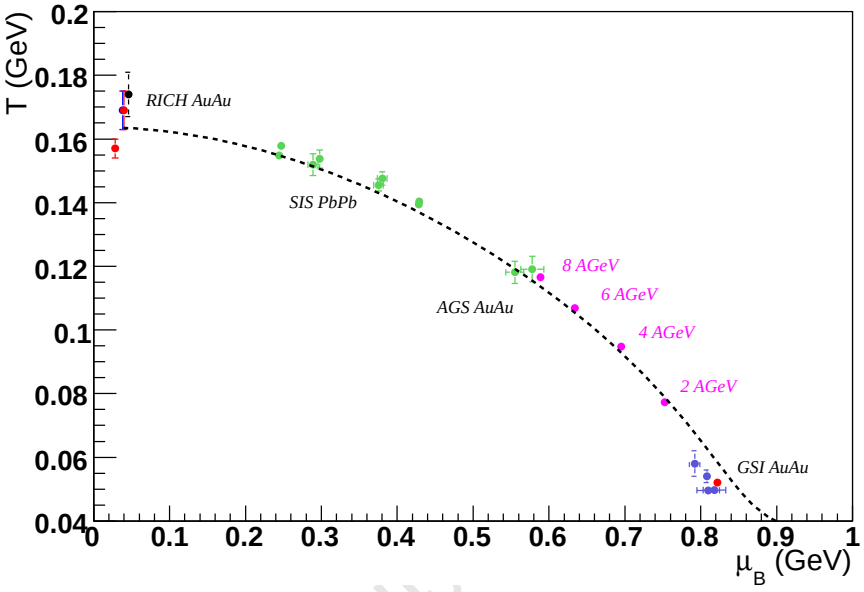


Figure 6.1 – A plot of fitted freeze-out points against the $\frac{E}{N} = 1$ GeV criterion which is indicated by the dashed line. The statistical-thermal fits to the data in Table 6.1 are shown in magenta.

statistical-thermal fits to the data. As is evident from the figure, the new data points agree very well with the $E/N = 1$ GeV line.

A set of preliminary pion yields [141] were produced which differe from those in [139]. There was a disagreement over the normalisation used. These yields were quoted with yields for Λ and K^0 and are listed in Table 6.3. The data values together with

E_{beam} (AGeV)	Λ	K^0	π^+	π^-
2	0.8±0.2	0.6±0.2	23±1	40±2
4	8.1±0.6	7±2	71±4	99±5
6	12.8±0.6	16±2	110±6	133±8
8	21±4	21±6	155±13	185±17

Table 6.3 – Preliminary 4π data [141] from central Au-Au collisions with nominal beam energies 2, 4, 6 and 8 AGeV at AGS. For all energies $N_{part} = 340 \pm 34$

$N_{part} = 340 \pm 34$ were also fitted using the mixed-canonical ensemble with the correlation volume assumed to be equal to the freeze-out volume and the charge chemical potential μ_Q constrained such that $B/2Q = 1.24662$. The fit values, obtained from THERMUS, are presented in Table 6.4 and are plotted in Fig 6.2.

Energy AGeV	T MeV	μ_B MeV	γ_S	R fm	χ^2
8	138.6±12.2	496.3±29.7	0.63±0.12	7.53±1.54	0.886
6	121.5±0.5	554.5±28.8	0.50±0.03	8.55±0.34	16.837
4	112.4±0.4	662.3±1.1	0.442±0.001	7.70± 10.55	2.702
2	79.2±0.4	781.3±0.2	0.21±0.03	9.16±0.05	2.0833

Table 6.4 – THERMUS fit to 4π data in Table 6.3.

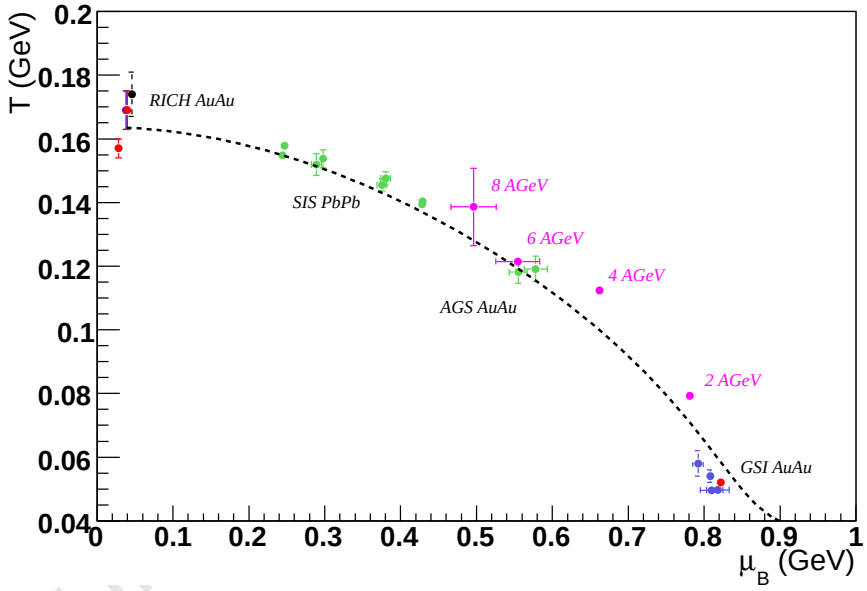


Figure 6.2 – A plot of fitted freeze-out points against the $\frac{E}{N} = 1$ GeV criterion which is indicated by the dashed line. The statistical-thermal fits to the data in Table 6.3 are shown in magenta.

It is evident from Fig 6.1 and Fig 6.2 that the temperatures for these alternative yields are much higher than for the previous results. This is due to the pion yields being significantly higher. In fact for the 6 and 8 AGeV points the temperatures are higher and the chemical potentials lower than the 11.6 AGeV point (lowest green point). This leads us to believe that the original yields, listed in Table 6.1, are more likely the correct values.

Chapter 7

Conclusion

The purpose of writing is to inflate weak ideas, obscure pure reasoning, and inhibit clarity. With a little practise, writing can be an intimidating and impenetrable fog!

– Bill Watterson (Calvin and Hobbes)

It is important to take into account hydrodynamic effects when trying to understand many of the observables in heavy ion collisions. To this end the package FCTHydro was developed. FCTHydro was designed to run in ROOT, the primary framework for analysing heavy ion collisions. The goal is to make hydrodynamic simulations more accessable to the broader heavy ion collision community with the hope that they will become more widely applied.

ROOT applications

The flux corrected transport package FCTHydro is an implementation of the LCPFCT algorithm [97] designed for solving generalised continuity equations. The algorithm was generalised slightly so as to solve conservative continuity equations of the form

$$\frac{\partial \rho}{\partial t} + \frac{1}{r^{\alpha-1}}\frac{\partial}{\partial r}(r^{\alpha-1}\rho v) = C_1\frac{1}{r^{\alpha-1}}\frac{\partial}{\partial r}(r^{\alpha-1}D_1) + C_2\frac{\partial D_2}{\partial r} + D_3, \tag{7.1}$$

and non-conservative continuity equations of the form

$$\frac{\partial \rho}{\partial t} + v\frac{1}{r^{\alpha-1}}\frac{\partial}{\partial r}(r^{\alpha-1}\rho) = C_1\frac{1}{r^{\alpha-1}}\frac{\partial}{\partial r}(r^{\alpha-1}D_1) + C_2\frac{\partial D_2}{\partial r} + D_3, \tag{7.2}$$

for Cartesian ($\alpha = 1$), cylindrical ($\alpha = 2$) and spherical ($\alpha = 3$) geometries. Since it is an implementation of the LCPFCT algorithm it is a high-order, monotone, conservative, positivity preserving algorithm and has inherited LCPFCT strengths and versatility. It has the ability to treat steep gradients and shockwaves, can be applied to Cartesian, polar and spherical geometries, as well as applied to both Eulerian and Lagrangian grids and has forth order phase accuracy for certain systems. There are also parameters which can be altered to fine tune the numerics.

In order to import FCTHydro into ROOT, it has been coded in C++ and compiled into a shared library, which can loaded into ROOT when needed. By using C++, FCTHydro has access to the advantages of this programming language. For example, FCTHydro has been designed with a dynamic system size functionality. The size of the system can be specified at runtime, as opposed to being hard coded and the application recompiled each time the system size changes.

The tests which were performed in chapter 2 served as a consistency check, confirming that the algorithm was implemented correctly and showing the behaviour of the numerical solutions in the various geometries. They also show how accurate FCTHydro is with a relative error of the order of 10^{-15} in the integration of the densities before and after 75 timesteps, for the conservative convection.

Since FCTHydro actually simulates continuity equations, it is used as a tool within other programs/ROOT macros to simulate hydrodynamic systems. RelHydro, another ROOT application, was developed to illustrate the application of FCTHydro to relativistic hydrodynamic systems, in particular, to (1+1) dimensional relativistic hydrodynamic systems. RelHydro is designed to simulate a system with any equation of state with the rules for creating one's own equation of state class described in chapter 3. However, there are four different equation of state classes packaged with RelHydro: a massless gas, a thermal gas of hadrons, a tabulated equation of state, and one for a quark-gluon gas which has a first order phase transition to a pion gas.

The shock wave tests showed the accuracy of RelHydro, and thus FCTHydro, in simulating relativistic systems, even those with steep gradients. By varying the parameters the following were identified as necessary for the best simulations:

- The two step method should be employed.
- The ratio $\frac{\mathrm{d}t}{\mathrm{d}x}$ should be set such that the courant number is close to, but not greater than, $\frac{1}{2}$.
- The antidiffusive fluxes should be corrected.
- The added diffusion must be fine-tuned to achieve a balance between the effects of dispersion and the added diffusion.

RelHydro was also applied to the well known initial condition of Landau [32] which consists of a static square wave. This system involves shock wave expansions into a vacuum as well as a region of interference when the two rarefaction waves from the edges meet in the centre. Comparing the freezeout curve (an isotherm) found by RelHydro with the analytic solution shows that the numerical simulation behaves extremely well even when subjected to these harsh initial conditions. The fact that temperature of the numerical solution cools slightly slower than the analytic solution is understandable as one cannot have an exact square wave initial condition in numerics. The gradients of a true square wave are larger than those of the numerical approximation and, as such, the analytic system has a higher acceleration away from the centre, and cools faster. This explanation is strengthened by the comparison of the cooling rates from an initial square wave distribution with those from an initial Woods-Saxon distribution.

Perhaps the most useful equation of state class which comes with RelHydro is the tabulated equation of state. This class admits a number density - energy density grid, on which the parameters of the system are defined, with an irregular spacing. It is important to check that the grid spacing, when tabulating an equation of state, is sufficiently fine. Otherwise, the errors in the interpolated thermodynamic parameters will be too big.

Not only does RelHydro illustrate an application of FCTHydro to relativistic systems, but it is, in its own right, a useful tool. Many first order corrections can be made using simple (1+1) dimensional hydrodynamic simulations.

Since our interest lies in heavy ion collisions, an ultimate goal would be to introduce second order non-ideal hydrodynamic corrections to simulations of the expansion of the system after collision. Chapter 4 explains how these systems can be simulated using FCTHydro. Before studying the complicated, full system, a better understanding of causal non-ideal systems is required by analysing them in the weak coupling and non-relativistic limit. The package NonIdeal_xy was developed to simulate this scenario in (2+1) dimensions.

The NonIdeal_xy package also provides an illustration on how FCTHydro can be applied to multi-dimensional systems as well as causal non-ideal systems. It is for the simulation of causal non-ideal systems that the non-conservative continuity equation is encountered since it is required for the evolution of the stress tensor components.

NonIdeal_xy was applied to a Boltzmann gas of Israel particles with heat flux switched off. This gave qualitative behaviours for the viscosity coefficient η and its relaxation time τ_π. The investigation of these parameters through the adjustment of their magnitudes via multiplicative factors (set by the parameters $eta0$ and $tau0$) found the following:

- As τ_π is altered, by changing the multiplicative factor $tau0$, a "phase change" occurs
- For values of $tau0$ one order and more below that of the phase change's value the distributions are very similar and are close to that of an ideal hydrodynamic system
- For values one order and more above the phase change's the distributions are also very similar to each other
- As τ_π is increased the initial conditions for the stress tensor components become less significant.

It was also found that if the value of $eta0$ is increased beyond 10^7, the system initially expands but then contracts in the centre. This behaviour is feasible as these values corresponds to extremely viscous systems $\eta \gtrsim 3 \times 10^2\ \mathrm{g\,cm^{-1}s^{-1}}$.

The effect of the viscosity coefficient, η, and its relaxation time, τ_π, on the eccentricity of the system was also investigated. It was found that although the distributions diverge from ideal system's distributions as τ_π increases, the evolution of the eccentricity diverges further from the ideal system's eccentricity evolution as τ_π decreases. The results also showed that as η is increased, the evolution of the eccentricity decreases ever slower. If the viscosity is increased too far, the eccentricity begins to increase after an initial decrease.

The values of viscosity for which this occurs correspond to same values described above for which the systems begin to contract after the initial expansion.

There was a complication when dealing with these non-relativistic systems, because the value of τ_π given by a Boltzmann gas of Israel particles suggested a much smaller step-size, dt, than was suggested by the courant number. For this reason the package NonIdeal_xy_pi10 was developed. This package evolves the stress components 10 times over a step-size $\frac{dt}{10}$ to a single evolution of the energy, momenta and density over a single step dt. This finer granularity in the evolution of the stress terms only made a significant difference for high values of $\eta \gtrsim 30\ \mathrm{g\,cm^{-1}s^{-1}}$ ($eta_0 \geq 10^6$). Fortunately, it is shown that for systems expected in heavy ion collisions, the proposed values for dt from τ_π and the courant number do not display such a large discrepancy.

Semi-analytic Landau spectra

In chapter 5 a semi-analytic solution for the $\frac{\mathrm{d}N}{\mathrm{d}y}$ spectra from (1+1) dimensional relativistic hydrodynamics with Landau's initial conditions is presented. This was developed as it was shown in chapter 3 that numerical simulations of the freezeout curve, for Landau's system, are not feasible when the ratio of the freezeout temperature to that of the initial temperature is of the order of 10^{-1} and smaller. The appearance of the double peak discovered previously in a numerical simulation is shown to disappear as the ratio of temperatures mentioned above diminishes. The results also confirm that the approximation devised in [117] is accurate for central rapidities.

E/N Freezeout Criteria

Finally in the last chapter the freezeout criteria are found for preliminary data [119] from AGS for Au-Au collisions at nominal beam energies of 2, 4, 6 and 8 AGeV. The results fill in previously unknown regions of the temperature - chemical potential plane and are consistent with the Cleymans-Redlich $E/N = 1$ GeV freezeout condition. This purely phenomenological model continues to describe the chemical freezeout points incredibly well. These results were used in [142] Fig 1.

Future Directions

The future directions for this work lie in the applications of FCTHydro to hydrodynamic systems. It would be useful to create (2+1) and (3+1) dimensional versions of the application Relhydro. Moreover, these higher dimensional applications should contain the appropriate freezeout mechanisms. On the side of the causal non-ideal hydrodynamic applications there is lots of room for development. The next move should, perhaps, be to apply FCTHydro to the full relativistic system in (2+1) dimensions and then to (3+1) dimensional systems. Another avenue for the future would be a test to determine how the use of a Lagrangian grid system compares to that of an Eulerian one. Parallelisation of the hydrodynamic code would allow for the simulation of larger systems and or systems with finer grids. This however, is a fairly complicated wish.

Appendix A

Kinematics

Unless specified otherwise, we have made use of natural units throughout this thesis:

$$\hbar = c = k = 1.$$

In a heavy ion collision, the beam axis (z-axis) defines a unique and common direction for all collisions in an experiment, while the x- and y-axes are determined on an event by event basis. Therefore, observables are considered in terms of the longitudinal momentum (p_z) and the transverse momentum ($p_\perp = \sqrt{p_x^2 + p_y^2}$). Since the collision energies are so large, it is convenient to introduce the variables longitudinal rapidity and transverse mass. The longitudinal rapidity of a particle with energy E is

$$y \equiv \tanh^{-1}\left(\frac{p_z}{E}\right)$$

and if the rest mass is m_0, the transverse mass is

$$m_\perp \equiv \sqrt{m_0^2 + p_\perp^2}. \tag{A.1}$$

In terms of these variables

$$\begin{aligned} E &= m_\perp \cosh y, \\ p_z &= m_\perp \sinh y, \\ p_x &= p_\perp \cos\phi, \\ p_y &= p_\perp \sin\phi, \end{aligned}$$

with ϕ the polar angle of the particle in the transverse plane. The four-momentum $p^\nu = [E, p_x, p_y, p_z]$ in terms of these new variables is

$$p^\nu = [m_\perp \cosh y, p_\perp \cos\phi, p_\perp \sin\phi, m_\perp \sinh y]. \tag{A.2}$$

The differentail momentum element $\mathrm{d}^3p = \mathrm{d}p_z\,\mathrm{d}p_x\,\mathrm{d}p_y$ can also be recast in terms of the differentials $\mathrm{d}y$, $\mathrm{d}m_\perp$ and $\mathrm{d}\phi$. To do this we first realise that $p_\perp = \sqrt{m_\perp^2 - m_0^2}$

and make use of the Jacobian:

$$\begin{aligned}
\mathrm{d}^3p &= \frac{\partial(p_x, p_y, p_z)}{\partial(y, m_\perp, \phi)}\,\mathrm{d}y\,\mathrm{d}m_\perp\,\mathrm{d}\phi \\
&= \begin{vmatrix} \partial_y p_x & \partial_{m_\perp} p_x & \partial_\phi p_x \\ \partial_y p_y & \partial_{m_\perp} p_y & \partial_\phi p_y \\ \partial_y p_z & \partial_{m_\perp} p_z & \partial_\phi p_z \end{vmatrix} \mathrm{d}y\,\mathrm{d}m_\perp\,\mathrm{d}\phi \\
&= \begin{vmatrix} 0 & \frac{m_\perp}{p_\perp}\cos\phi & -p_\perp\sin\phi \\ 0 & \frac{m_\perp}{p_\perp}\sin\phi & p_\perp\cos\phi \\ m_\perp\cosh y & \sinh y & 0 \end{vmatrix} \mathrm{d}y\,\mathrm{d}m_\perp\,\mathrm{d}\phi
\end{aligned}$$

and so

$$\mathrm{d}^3p = Em_\perp\,\mathrm{d}y\,\mathrm{d}m_\perp\,\mathrm{d}\phi. \tag{A.3}$$

This can also be expressed in terms of the the differentials $\mathrm{d}y$, $\mathrm{d}p_\perp$ and $\mathrm{d}\phi$ as

$$\mathrm{d}^3p = Ep_\perp\,\mathrm{d}y\,\mathrm{d}p_\perp\,\mathrm{d}\phi. \tag{A.4}$$

since $p_\perp\,\mathrm{d}p_\perp = m_\perp\,\mathrm{d}m_\perp$, from equation (A.1).

The rapidity spectra of an observable, $\mathcal{O}$, can then be obtained via

$$\frac{\mathrm{d}\mathcal{O}}{\mathrm{d}y} = \int_0^{2\pi}\int_0^{\infty} \frac{\mathrm{d}^3\mathcal{O}}{\mathrm{d}p^3} Ep_\perp\,\mathrm{d}p_\perp\,\mathrm{d}\phi \tag{A.5}$$

$$= \int_0^{2\pi}\int_{m_0}^{\infty} \frac{\mathrm{d}^3\mathcal{O}}{\mathrm{d}p^3} Em_\perp\,\mathrm{d}m_\perp\,\mathrm{d}\phi \tag{A.6}$$

Appendix B

Configuration File Example

This is an example of a configuration file for **TRelHydro**. Notice that for each variable, its name is followed by an "=" sign buffered by blank spaces and then its value. Anything which occurs after a "#" is ignored, which allows one to add comments to the configuration file. Blank lines are also ignored. If variables are set in the configuration file but not needed by the specific application of **TRelHydro**, then they are ignored. For example if the massless particle equation of state is being used then the parameters for the quark-gluon model and thermal hadron model are ignored.

```
t0xmode = 2              # 2<+c*grad()>|5<+c*grad()> (from interface data)
t00mode = 1              # 1<+div()> |4<+div()> (from interface data)
twoStep = 1              # bool switch for twostep metod
scrhSwitch = 1.0         # bool switch fow scrh term
residualDiff = 0.9       # parameter which determines the amount of extra diffusion
correctAntidiffFlux = 1  # bool switch for correcting the antidiffusive fluxes
writeInterval = 500      # number of iterations between outputs
noWrites = 1             # number of outputs
dx = 0.01
dt = 0.005
N = 14001
x0 = -70.0
time = 0.0               # initial time
gridtype = E             # E - eulerian grid
alpha = 1                # 1-cartesian, 2-polar, 3-spherical
toll = 1.0e-16           # vacumme is < toll
Tfo = 0.25               # freezeout temperature [GeV]

# the next 2 are for qg Model eos
Bag = 0.0         # Bag model parameter
Nf = 2            # Number of flavours
Nc = 3            # Number of colours

# the next 4 are for WS initialisation, if used
a = 1.0           # skin width of wood-saxons dist = a*dx
e0 = 43.0308388   # maximum energy density
b0 = 0.5          # maximum baryon density
r0 = 1.1          # half width of distribution
```

```
# the next 5 are for the Thermal Model eos (needed by THERMUS)
withWidths = 0              # 0-false 1-true
qStats = 1                  # 0-false 1-true
plist1 = ./PartList.txt
plist2 = ./PartList.txt
bFitEpsCut = 1e-16
```

Appendix C

Filling a TRHTabEOS Object

If you have a tabulated version of an equation of state which you wish to use with **TRelHydro**, it must be contained in a **TRHTabEOS** object. Here we present two macros showing how one can fill a **TRHTabEOS** object. In the first macro, matrices are created using the Massless Particle equation of state (see sections §3.1.1 and §3.2.2). In the second macro these matrices are used to fill the **TRHTabEOS** object.

The macro for filling the matrices with the values of the pressure, temperature, etc:

```
{
  //in this macro e is energy density and nb is baryon density

  gSystem->Load("../lib/libFCTHydro2.so");    // load FCTHydro library
  gSystem->Load("../lib/libRHTabEos2.so");    // load TRHTabEOS library
  gSystem->Load("../lib/libRHMassless2.so");  // load TRHMassless library

  TRHMassless bob;      // Initialise a TRHMassless object
  bob.SetVectorSize(1); // Set size to one as we will only need one point
                        //   for each calculation

  Int_t ebins=100,nbins=100;    // number of e and nb bins
  Double_t eLow=0.0,eHigh=1.0;  // low and high e values
  Double_t nLow=0.0,nHigh=1.0;  // low and high nb values
  Double_t de=(eHigh-eLow)/((Double_t)(ebins-1));
  Double_t dn=(nHigh-nLow)/((Double_t)(nbins-1));
  Double_t eDens[ebins], nDens[nbins]; // arrays for e and nb

  // the following are matrices which will contain the values of the
  //  temperature, pressure, MuB, MuS, entropy and number density
  //  TMatrixD objects are instantiated with zeroed entries
  TMatrixD hT(ebins,nbins), hP(ebins,nbins);
  TMatrixD hMb(ebins,nbins), hMs(ebins,nbins);
  TMatrixD hEnt(ebins,nbins), hNum(ebins,nbins);

  Double_t Eps[2],Nb[2],pres[2]; // vectors for using the InitPresTemp()
                                 //  routine of TRHMassless
```

```
  // These loops fill the [i][j]'th components of the matrices with values
  //   corresponding to eDens[i] and nDens[j];
  for (Int_t i=0;i<=ebins-1;i++)
  {
    eDens[i]=eLow +((Double_t)(i))*de;
    Eps[1] =  eDens[i];
    Int_t broken = 0;
    for (Int_t j=0;j<=nbins-1;j++)
    {
      nDens[j]=nLow +((Double_t)(j))*dn;
      Nb[1] =  nDens[j];
      bob.InitPresTemp(Eps,Nb,pres);
      hT[i][j]=bob.Temp[1];
      hP[i][j]=pres[1];
      hEnt[i][j]=bob.GetEntropy(1,Eps,Nb,pres);
      hNum[i][j]=bob.GetNumDens(1,Eps,Nb,pres);
    }
  }
}
```

The following macro fills the **TRHTabEOS** object with the values in the matrices. The columns of the matrices, which correspond to fixed values of the baryon density, are inputted into separate **BDensObj** objects. These objects are then added to the **TRHTabEOS** object.

```
{
  TRHTabEOS tom;      // create TRHTabEOS object
  BDensObj* bvec;
  bvec = new BDensObj [nbins]; // create BDensObj array with
                               //   same number of bins as nb

  for (Int_t i=0;i<=nbins-1;i++)
  {
    (bvec[i]).SetSize(ebins);   // set i'th BDensObj to have ebins
                                //   elements in all its arrays;
    (bvec[i]).density=nDens[i]; // set density value to that of
                                //   i'th element of nb

    // The energy density, pressure, temperature, MuB, MuS
    //   and entropy arrays are now filled with corresponding
    //   values from the matrices.
    for (Int_t j=0;j<ebins;j++)
    {
      (bvec[i]).Edens[k]=eDens[j];
      (bvec[i]).Pres[k]=hP[j][i];
      (bvec[i]).T[k]=hT[j][i];
      (bvec[i]).Mub[k]=hMb[j][i];
      (bvec[i]).Mus[k]=hMs[j][i];
      (bvec[i]).Ent[k]=hEnt[j][i];
       k++;
    }
```

```
    tom.Bdens.Add(&(bvec[i])); //the i'th BdensObject is added
                               //  to the object array in the
                               //  TRHTabEOS object
  }
}
```

Now that the **TRHTabEOS** object has been created it can be save in a ROOT file and then loaded when required by **TRelHydro**.

Note!

With the manner in which **TRHTabEOS** has been coded, the size of the arrays for the different **BDensObj** objects do not have to be the same. Thus, recalling that $bvec[i].Ebins$ is the size of the arrays in the i'th **BDensObj** object in $bvec$, we have that $bvec[i].Ebins$ is not necessarily equal to $bvec[k].Ebins$, if $i \neq k$. However it is important to realise that the corresponding energy density elements, if they exist, must be equal. For example, if $bvec[i].Edens[j]$ and $bvec[k].Edens[i]$ exist, then $bvec[i].Edens[j] = bvec[k].Edens[i]$. The element $bvec[k].Edens[i]$ will exist if $i \leq vec[k].Ebins$.

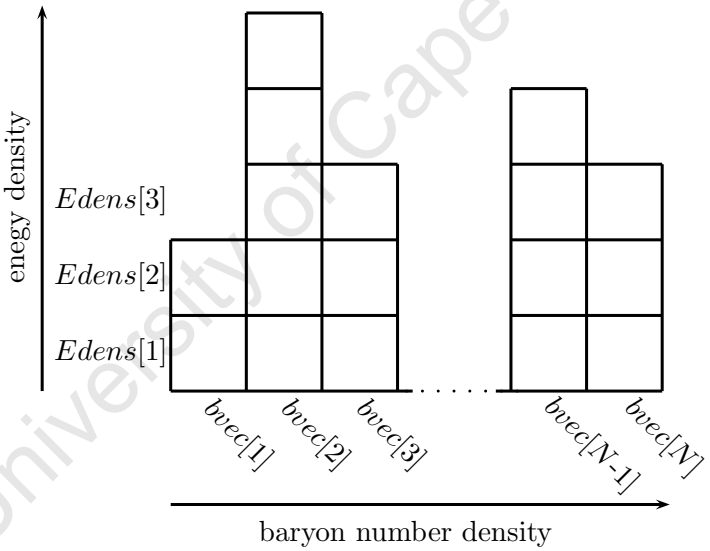


Figure C.1 – Illustration of the allowed setup of a **TRHTabEOS** object. Each column corresponds to a **BdensObject** in the array $bvec$. Each row corresponds to an element in the $Edens$ arrays in each $bvec$ element. To each $bvec$ element there may be a different number of rows, but corresponding rows have the same energy density value. For each cell in this grid the pressure, temperature, μ_B, μ_S and entropy for that value of baryon number density and energy density are stored.

Appendix D

TNonideal_xy_pi10

This package is used to simulate the same systems as **TNonideal_xy** (§4.2). However, **TNonideal_xy_pi10** evolves the stress tensor componets forward through ten timesteps of stepsize $dt/10$ to each evolution of the energy, momenta and number density evolutions forward a single timestep of stepsize dt. There are two classes, the main class **TNonideal_xy_pi10** and a container class **TNIParam**. This container class has exactly the same parameters and functions as the one for **TNonideal_xy**. See section §4.2.1 for a description its the parameters and functions.

The the main class for simulating the system has a similar form to that of **TNonideal_xy** and has the four parameters

xFluid	**FCTFluid**	object for doing the x-direction 1D evolutions
yFluid	**FCTFluid**	object for doing the y-direction 1D evolutions
xvelocity	**FCTVeloctiy**	object for calculating the velocity dependent parameters for x-direction 1D evolutions
yvelocity	**FCTVeloctiy**	object for calculating the velocity dependent parameters for y-direction 1D evolutions

in place of the two parameters *velocity* and *Fluid* of **TNonideal_xy_pi10**.

The functions of **TNonideal_xy_pi10** which do not appear in **TNonideal_xy** are the private functions

ConvectT00(Double_t *valdt*)	Int_t
ConvectPIsX(Double_t *valdt*)	Int_t
SetSourceVP()	void
SetSourcePIs()	void

The function ConvectT00() is used for convecting the energy, momentum and number density distributions forward one timestep *valdt*. ConvectPIsX() convects the stress tensor,bulk stress and heat flux forward one timestep *valdt*, in the x-direction. ConvectPIsY() performs the similar function, but in the y-direction. These replace the functions ConvectDensityX() and ConvectDensityY() of **TNonideal_xy**. SetSourceVP() determines the values of the pressure and velocity components which are needed for calculating source terms for the evolution equations. Similarly SetSourcePIs() calculates the stress and heat flux parameters needed for the evolution equations. These replace the function SetSourceValues() of **TNonideal_xy**. The reason for the introduction of the new functions is to faciliate the separate simulation of the stress tensor and that of the energy, momentum and number densities.

The functions of **TNonideal_xy_pi10** that are called to simulate a system have the same names and usage as those of **TNonideal_xy**.

Bibliography

[1] R. P. Feynman, "Very High-Energy Collisions of Hadrons," *Phys. Rev. Lett.* **23** (Dec, 1969) 1415–1417.

[2] H. Fritzsch, M. Gell-Mann, and H. Leutwyler, "Advantages of the Color Octet Gluon Picture," *Phys. Lett.* **B47** (1973) 365–368.

[3] **Particle Data Group** Collaboration, W. M. Yao *et al.*, "Review of particle physics," *J. Phys.* **G33** (2006) 1–1232.

[4] S. Weinberg, "A Model of Leptons," *Phys. Rev. Lett.* **19** (1967) 1264–1266.

[5] A. Salam, "Weak and Electromagnetic Interactions,". Originally printed in *Svartholm: Elementary Particle Theory, Proceedings Of The Nobel Symposium Held 1968 At Lerum, Sweden*, Stockholm 1968, 367-377.

[6] D. J. Gross and F. Wilczek, "Asymptotically Free Gauge Theories. 1," *Phys. Rev.* **D8** (1973) 3633–3652.

[7] D. J. Gross and F. Wilczek, "Ultraviolet Behavior of Non-Abelian Gauge Theories," *Phys. Rev. Lett.* **30** (1973) 1343–1346.

[8] H. D. Politzer, "Reliable Perturbative Results for Strong Interactions?," *Phys. Rev. Lett.* **30** (1973) 1346–1349.

[9] H. D. Politzer, "Asymptotic Freedom: An Approach to Strong Interactions," *Phys. Rept.* **14** (1974) 129–180.

[10] F. Karsch, "Lattice results on QCD thermodynamics," *Nucl. Phys.* **A698** (2002) 199–208, `hep-ph/0103314`.

[11] F. Karsch, E. Laermann, and A. Peikert, "Quark mass and flavor dependence of the QCD phase transition," *Nucl. Phys.* **B605** (2001) 579–599, `hep-lat/0012023`.

[12] **MILC** Collaboration, C. Bernard *et al.*, "QCD thermodynamics with three flavors of improved staggered quarks," *Phys. Rev.* **D71** (2005) 034504, `hep-lat/0405029`.

[13] J. C. Collins and M. J. Perry, "Superdense Matter: Neutrons Or Asymptotically Free Quarks?," *Phys. Rev. Lett.* **34** (1975) 1353.

[14] H. Reeves, "Big Bang nucleosynthesis and the quark - hadron phase transition," *Phys. Rept.* **201** (1991) 335–354.

[15] N. K. Glendenning, S. Pei, and F. Weber, "Signal of quark deconfinement in the timing structure of pulsar spin-down," *Phys. Rev. Lett.* **79** (1997) 1603–1606, `astro-ph/9705235`.

[16] G. F. Chapline, M. H. Johnson, E. Teller, and M. S. Weiss, "Highly excited nuclear matter," *Phys. Rev.* **D8** (1973) 4302–4308.

[17] P. Braun-Munzinger, K. Redlich, and J. Stachel, "Particle production in heavy ion collisions," `nucl-th/0304013`.

[18] K. Redlich, J. Cleymans, H. Oeschler, and A. Tounsi, "Particle production and equilibration in heavy ion collisions," *Acta Phys. Polon.* **B33** (2002) 1609–1628.

[19] R. Baier, A. H. Mueller, D. T. Son, and D. Schiff, "Parton thermalization and energy loss," *Nucl. Phys.* **A698** (2002) 217–226.

[20] R. Baier, A. H. Mueller, D. Schiff, and D. T. Son, "'Bottom-up' thermalization in heavy ion collisions," *Phys. Lett.* **B502** (2001) 51–58, `hep-ph/0009237`.

[21] R. J. Glauber, *Lectures on theoretical Physics*, vol. I. Inter-Science, New-York, 1959.

[22] A. Bialas and W. Czyz, "Overlap Function in High-Energy Collisions of Composite Objects," *Acta Phys. Polon.* **B7** (1976) 779.

[23] W. Broniowski and W. Florkowski, "Geometric relation between centrality and the impact parameter in relativistic heavy ion collisions," *Phys. Rev.* **C65** (2002) 024905, `nucl-th/0110020`.

[24] K. Kajantie, P. V. Landshoff, and J. Lindfors, "Minijet Production in High-Energy Nucleus-Nucleus Collisions," *Phys. Rev. Lett.* **59** (1987) 2527.

[25] K. J. Eskola, K. Kajantie, and J. Lindfors, "Quark and Gluon Production in High-Energy Nucleus-Nucleus Collisions," *Nucl. Phys.* **B323** (1989) 37.

[26] X.-N. Wang and M. Gyulassy, "HIJING: A Monte Carlo model for multiple jet production in p p, p A and A A collisions," *Phys. Rev.* **D44** (1991) 3501–3516.

[27] K. Geiger, "VNI 3.1: MC-simulation program to study high-energy particle collisions in QCD by space-time evolution of parton-cascades and parton-hadron conversion," *Comput. Phys. Commun.* **104** (1997) 70–160, `hep-ph/9701226`.

[28] B. Zhang, "ZPC 1.0.1: A parton cascade for ultrarelativistic heavy ion collisions," *Comput. Phys. Commun.* **109** (1998) 193–206, `nucl-th/9709009`.

[29] S. A. Bass *et al.*, "Microscopic models for ultrarelativistic heavy ion collisions," *Prog. Part. Nucl. Phys.* **41** (1998) 255–369, `nucl-th/9803035`.

[30] B.-A. Li and C. M. Ko, "Formation of superdense hadronic matter in high-energy heavy ion collisions," *Phys. Rev.* **C52** (1995) 2037–2063, `nucl-th/9505016`.

[31] B. Zhang, C. M. Ko, B.-A. Li, and Z.-w. Lin, "A multi-phase transport model for nuclear collisions at RHIC," *Phys. Rev.* **C61** (2000) 067901, `nucl-th/9907017`.

[32] S. Z. Belenkij and L. D. Landau, "Hydrodynamic theory of multiple production of particles," *Nuovo Cim. Suppl.* **3S10** (1956) 15.

[33] **WA98** Collaboration, M. M. Aggarwal *et al.*, "Scaling of particle and transverse energy production in 208-Pb + 208-Pb collisions at 158-A-GeV," *Eur. Phys. J.* **C18** (2001) 651–663, `nucl-ex/0008004`.

[34] T. J. Hallman, D. E. Kharzeev, J. T. Mitchell, and T. Ullrich, eds., *15th International Conference On Ultrarelativistic Nucleus-Nucleus Collisions (QM2001)*, vol. A698. Nucl. Phys., Stony Brook, New York, 15-20 Jan, 2002.

[35] H. Gutbrod, J. Aichelin, and W. K., eds., *16th International Conference On Ultrarelativistic Nucleus-Nucleus Collisions (QM2002)*, vol. A715. Nucl. Phys., Nantes, France, 18-24 July, 2003.

[36] P. F. Kolb, J. Sollfrank, and U. W. Heinz, "Anisotropic transverse flow and the quark-hadron phase transition," *Phys. Rev.* **C62** (2000) 054909, `hep-ph/0006129`.

[37] P. F. Kolb, P. Huovinen, U. W. Heinz, and H. Heiselberg, "Elliptic flow at SPS and RHIC: From kinetic transport to hydrodynamics," *Phys. Lett.* **B500** (2001) 232–240, `hep-ph/0012137`.

[38] P. Huovinen, P. F. Kolb, U. W. Heinz, P. V. Ruuskanen, and S. A. Voloshin, "Radial and elliptic flow at RHIC: Further predictions," *Phys. Lett.* **B503** (2001) 58–64, `hep-ph/0101136`.

[39] P. F. Kolb, U. W. Heinz, P. Huovinen, K. J. Eskola, and K. Tuominen, "Centrality dependence of multiplicity, transverse energy, and elliptic flow from hydrodynamics," *Nucl. Phys.* **A696** (2001) 197–215, `hep-ph/0103234`.

[40] H. Stoecker and W. Greiner, "High-Energy Heavy Ion Collisions: Probing the Equation of State of Highly Excited Hadronic Matter," *Phys. Rept.* **137** (1986) 277–392.

[41] R. B. Clare and D. Strottman, "Relativistic Hydrodynamics and heavy ion reactions," *Phys. Rept.* **141** (1986) 177–280.

[42] J.-P. Blaizot and J.-Y. Ollitrault, "Hydrodynamics of Quark - Gluon Plasmas," *Adv. Ser. Direct. High Energy Phys.* **6** (1990) 393–470.

[43] P. F. Kolb and U. W. Heinz, "Hydrodynamic description of ultrarelativistic heavy-ion collisions," `nucl-th/0305084`.

[44] P. Huovinen, "Hydrodynamical description of collective flow," `nucl-th/0305064`.

[45] P. V. Ruuskanen, "Transverse Hydrodynamics with a First Order Phase Transition in very High-Energy Nuclear Collisions," *Acta Phys. Polon.* **B18** (1987) 551.

[46] L. P. Csernai, *Introduction to relativistic heavy ion collisions*. Wiley, Chichester, UK, 1994.

[47] D. H. Rischke, "Fluid dynamics for relativistic nuclear collisions," `nucl-th/9809044`.

[48] C. Eckart, "The Thermodynamics of irreversible processes. 3. Relativistic theory of the simple fluid," *Phys. Rev.* **58** (1940) 919–924.

[49] L. D. Landau and E. M. Lifshitz, *Course of Theoretical Physics: vol 6 Fluid Mechanics*. Pergamon Press, Oxford, 1963.

[50] A. Dumitru and D. H. Rischke, "Collective dynamics in highly relativistic heavy-ion collisions," *Phys. Rev.* **C59** (1999) 354–363, `nucl-th/9806003`.

[51] J.-Y. Ollitrault, "Anisotropy as a signature of transverse collective flow," *Phys. Rev.* **D46** (1992) 229–245.

[52] D. Teaney, J. Lauret, and E. V. Shuryak, "A hydrodynamic description of heavy ion collisions at the SPS and RHIC," `nucl-th/0110037`.

[53] D. Teaney, "Chemical freezeout in heavy ion collisions," `nucl-th/0204023`.

[54] T. Hirano, "Is early thermalization achieved only near midrapidity in Au + Au collisions at $\sqrt{s_{NN}} = 130$ GeV?," *Phys. Rev.* **C65** (2002) 011901, `nucl-th/0108004`.

[55] T. Hirano and K. Tsuda, "Collective flow and two pion correlations from a relativistic hydrodynamic model with early chemical freeze out," *Phys. Rev.* **C66** (2002) 054905, `nucl-th/0205043`.

[56] J. D. Bjorken, "Highly Relativistic Nucleus-Nucleus Collisions: The Central Rapidity Region," *Phys. Rev.* **D27** (1983) 140–151.

[57] U. Ornik, M. Plumer, B. R. Schlei, D. Strottman, and R. M. Weiner, "Hydrodynamical analysis of symmetric nucleus-nucleus collisions at CERN/SPS energies," *Phys. Rev.* **C54** (1996) 1381–1389, `hep-ph/9604323`.

[58] J. Sollfrank, P. Huovinen, and P. V. Ruuskanen, "Mass number scaling in ultra-relativistic nuclear collisions from a hydrodynamical approach," *Eur. Phys. J.* **C6** (1999) 525–536, `nucl-th/9801023`.

[59] K. Morita, S. Muroya, C. Nonaka, and T. Hirano, "Comparison of space-time evolutions of hot/dense matter in $\sqrt{s_{NN}} = 17$ GeV and 130 GeV relativistic heavy ion collisions based on a hydrodynamical model," *Phys. Rev.* **C66** (2002) 054904, `nucl-th/0205040`.

[60] E. Iancu and R. Venugopalan, "The color glass condensate and high energy scattering in QCD," `hep-ph/0303204`.

[61] V. K. Magas, L. P. Csernai, and D. D. Strottman, "The initial state of ultra-relativistic heavy ion collision," *Phys. Rev.* **C64** (2001) 014901, `hep-ph/0010307`.

[62] M. Gyulassy, D. H. Rischke, and B. Zhang, "Hot spots and turbulent initial conditions of quark-gluon plasmas in nuclear collisions," *Nucl. Phys.* **A613** (1997) 397–434, `nucl-th/9609030`.

[63] C. Nonaka, E. Honda, and S. Muroya, "(3+1)-dimensional relativistic hydrodynamical expansion of hot and dense matter in ultra-relativistic nuclear collision," *Eur. Phys. J.* **C17** (2000) 663–673, `hep-ph/0007187`.

[64] T. Osada, C. E. Aguiar, Y. Hama, and T. Kodama, "Event-by-event analysis of ultra-relativistic heavy-ion collisions in smoothed particle hydrodynamics," `nucl-th/0102011`.

[65] S. A. Bass *et al.*, "Hadronic freeze-out following a first order hadronization phase transition in ultrarelativistic heavy-ion collisions," *Phys. Rev.* **C60** (1999) 021902, `nucl-th/9902062`.

[66] S. A. Bass and A. Dumitru, "Dynamics of hot bulk QCD matter: From the quark-gluon plasma to hadronic freeze-out," *Phys. Rev.* **C61** (2000) 064909, `nucl-th/0001033`.

[67] S. Soff, S. A. Bass, and A. Dumitru, "Pion interferometry at RHIC: Probing a thermalized quark gluon plasma?," *Phys. Rev. Lett.* **86** (2001) 3981–3984, `nucl-th/0012085`.

[68] D. Teaney, J. Lauret, and E. V. Shuryak, "Flow at the SPS and RHIC as a quark gluon plasma signature," *Phys. Rev. Lett.* **86** (2001) 4783–4786, `nucl-th/0011058`.

[69] F. Cooper, G. Frye, and E. Schonberg, "Landau's Hydrodynamic Model of Particle Production and Electron Positron Annihilation into Hadrons," *Phys. Rev.* **D11** (1975) 192.

[70] K. A. Bugaev and M. I. Gorenstein, "Particle freeze-out in self-consistent relativistic hydrodynamics," `nucl-th/9903072`.

[71] K. A. Bugaev, "Shock-like Freeze-out in Relativistic Hydrodynamics," *Nucl. Phys.* **A606** (1996) 559–567, `nucl-th/9906047`.

[72] V. K. Magas *et al.*, "Freeze-out in hydrodynamical models in relativistic heavy ion collisions," *Nucl. Phys.* **A661** (1999) 596–599, `nucl-th/0001049`.

[73] E. Molnar, L. P. Csernai, V. K. Magas, A. Nyiri, and K. Tamosiunas, "Covariant description of kinetic freeze out through a finite space-like layer," *Phys. Rev.* **C74** (2006) 024907, `nucl-th/0503047`.

[74] E. Molnar *et al.*, "Covariant description of kinetic freeze out through a finite time-like layer," *J. Phys.* **G34** (2007) 1901–1916, nucl-th/0503048.

[75] A. M. Poskanzer and S. A. Voloshin, "Methods for analyzing anisotropic flow in relativistic nuclear collisions," *Phys. Rev.* **C58** (1998) 1671–1678, nucl-ex/9805001.

[76] P. F. Kolb, "v(4): A small, but sensitive observable for heavy ion collisions," *Phys. Rev.* **C68** (2003) 031902, nucl-th/0306081.

[77] **STAR** Collaboration, J. Adams *et al.*, "Azimuthal anisotropy at RHIC: The first and fourth harmonics," *Phys. Rev. Lett.* **92** (2004) 062301, nucl-ex/0310029.

[78] **STAR** Collaboration, C. Adler *et al.*, "Azimuthal anisotropy and correlations in the hard scattering regime at RHIC," *Phys. Rev. Lett.* **90** (2003) 032301, nucl-ex/0206006.

[79] P. Sorensen, "Evidence from identified particles for active quark and gluon degrees of freedom," *J. Phys.* **G32** (2006) S135–S141, nucl-ex/0701048.

[80] **STAR** Collaboration, M. D. Oldenburg, "Scaling of anisotropic flow in the picture of quark coalescence," *J. Phys.* **G31** (2005) S437–S442, nucl-ex/0412001.

[81] U. W. Heinz and P. F. Kolb, "Rapidity dependent momentum anisotropy at RHIC," *J. Phys.* **G30** (2004) S1229–S1234, nucl-th/0403044.

[82] I. Müller, "Zum Paradoxon der Wrmeleitungstheorie," *Z. Phys.* **198** (1967) 329–344.

[83] W. Israel and J. M. Stewart, "Transient relativistic thermodynamics and kinetic theory," *Ann. Phys.* **118** (1979) 341–372.

[84] H. C. Öttinger, "Relativistic and nonrelativistic description of fluids with anisotropic heat conduction," *Physica A* **254** (1998) 433–450.

[85] M. Grmela and H. C. Öttinger, "Dynamics and thermodynamics of complex fluids. I. Development of a general formalism," *Phys. Rev. E* **56** (Dec, 1997) 6620–6632.

[86] H. C. Öttinger and M. Grmela, "Dynamics and thermodynamics of complex fluids. II. Illustrations of a general formalism," *Phys. Rev. E* **56** (Dec, 1997) 6633–6655.

[87] D. Teaney, "Effect of shear viscosity on spectra, elliptic flow, and Hanbury Brown-Twiss radii," *Phys. Rev.* **C68** (2003) 034913, nucl-th/0301099.

[88] D. A. Teaney, "Viscosity and thermalization," *J. Phys.* **G30** (2004) S1247–S1250, nucl-th/0403053.

[89] A. Muronga, "Causal Theories of Dissipative Relativistic Fluid Dynamics for Nuclear Collisions," *Phys. Rev.* **C69** (2004) 034903, `nucl-th/0309055`.

[90] U. W. Heinz, H. Song, and A. K. Chaudhuri, "Dissipative hydrodynamics for viscous relativistic fluids," *Phys. Rev.* **C73** (2006) 034904, `nucl-th/0510014`.

[91] T. Koide, G. S. Denicol, P. Mota, and T. Kodama, "Relativistic dissipative hydrodynamics: A Minimal causal theory," *Phys. Rev.* **C75** (2007) 034909, `hep-ph/0609117`.

[92] R. Baier, P. Romatschke, and U. A. Wiedemann, "Dissipative hydrodynamics and heavy ion collisions," *Phys. Rev.* **C73** (2006) 064903, `hep-ph/0602249`.

[93] A. K. Chaudhuri, "Dissipative hydrodynamics in 2+1 dimension," *Phys. Rev.* **C74** (2006) 044904, `nucl-th/0604014`.

[94] H. Song and U. W. Heinz, "Suppression of elliptic flow in a minimally viscous quark- gluon plasma," *Phys. Lett.* **B658** (2008) 279–283, `0709.0742`.

[95] P. Romatschke and U. Romatschke, "Viscosity Information from Relativistic Nuclear Collisions: How Perfect is the Fluid Observed at RHIC?," *Phys. Rev. Lett.* **99** (2007) 172301, `0706.1522`.

[96] R. Brun and F. Rademakers, "ROOT: An object oriented data analysis framework," *Nucl. Instrum. Meth.* **A389** (1997) 81–86.

[97] J. Boris, A. Landsberg, E. Oran, and J. Gardner, "LCPFCT - A Flux-Corrected Transport Algorithm for Solving Generalized Continuity Equations.," *NRL Memorandum Report* **6410-93-7192** (1993).

[98] J. Boris and D. Book, "Flux-Corrected Transport 1. SHASTA, A Fluid Transport Algorithm That Works.," *J. Comput. Phys.* **11** (1973) 38.

[99] J. Boris, D. Book, and K. Hain, "Flux-Corrected Transport II: Generalizations of the Method," *J. Comput. Phys.* **18** (1975) 248–283.

[100] J. Boris and D. Book, "Solution of Continuity Equations by the Method of Flux-Corrected Transport," *Methods in Computational Physics* **16** (1976) 85–129.

[101] S. Wheaton and J. Cleymans, "THERMUS: A thermal model package for ROOT," `hep-ph/0407174`.

[102] J. Rafelski, "Strange anti-baryons from quark - gluon plasma," *Phys. Lett.* **B262** (1991) 333–340.

[103] P. Koch, B. Muller, and J. Rafelski, "Strangeness in Relativistic Heavy Ion Collisions," *Phys. Rept.* **142** (1986) 167–262.

[104] A. Chodos, R. L. Jaffe, K. Johnson, C. B. Thorn, and V. F. Weisskopf, "A New Extended Model of Hadrons," *Phys. Rev.* **D9** (1974) 3471–3495.

[105] S. Wheaton and J. Cleymans, "Statistical-thermal model calculations using THERMUS," *J. Phys.* **G31** (2005) S1069–S1074, hep-ph/0412031.

[106] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling, *Numerical Recipes in C.* Cambridge University Press, 1992. http://www.nrbook.com/b/.

[107] V. Schneider *et al.*, "New algorithms for ultrarelativistic numerical hydrodynamics," *J. Comput. Phys.* **105** (1993) 92–107.

[108] S. Bernard, J. A. Maruhn, W. Greiner, and D. H. Rischke, "Relativistic Hydrodynamics for Heavy–Ion Collisions: Freeze–Out and Particle Spectra," *Nucl. Phys.* **A605** (1996) 566–582, nucl-th/9602011.

[109] W. A. Hiscock and L. Lindblom, "Linear plane waves in dissipative relativistic fluids," *Phys. Rev. D* **35** (Jun, 1987) 3723–3732.

[110] W. A. Hiscock and L. Lindblom, "Generic instabilities in first-order dissipative relativistic fluid theories," *Phys. Rev. D* **31** (Feb, 1985) 725–733.

[111] H. Grad, "On the Kinetic Theory of Rarefied Gases," *Commun. Pure Appl. Math.* **2** (1949) 331–407.

[112] A. Muronga, "Relativistic Dynamics of Non-ideal Fluids: Viscous and heat-conducting fluids II. Transport properties and microscopic description of relativistic nuclear matter," *Phys. Rev.* **C76** (2007) 014910, nucl-th/0611091.

[113] A. Muronga, "Relativistic Dynamics of Non-ideal Fluids: Viscous and heat-conducting fluids I. General Aspects and 3+1 Formulation for Nuclear Collisions," *Phys. Rev.* **C76** (2007) 014909, nucl-th/0611090.

[114] W. Israel, "Relativistic Kinetic Theory of a Simple Gas," *Journal of Mathematical Physics* **4** (1963), no. 9, 1163–1181.

[115] S. Chapman and T. Cowling, *The mathematical theory of non-uniform gases, 3rd edition.* Cambridge University Press, Cambridge, 1970.

[116] S. de Groot, W. van Leeuwen, and C. van Weert, *Relativistic Kinetic Theory, Principles and Applications.* North-Holland Publishing Company, Amsterdam, 1980.

[117] D. K. Srivastava, J. Alam, and B. Sinha, "Rapidity distribution of secondaries in ultrarelativistic heavy ion collisions using Landau's hydrodynamic model," *Phys. Lett.* **B296** (1992) 11–17.

[118] **BRAHMS** Collaboration, I. G. Bearden *et al.*, "Charged meson rapidity distributions in central Au + Au collisions at $\sqrt{s_{NN}} = 200$ GeV," *Phys. Rev. Lett.* **94** (2005) 162301, nucl-ex/0403050.

[119] D. Cebra. private communication to J. Cleymans.

[120] E. Fermi, "High-energy nuclear events," *Prog. Theor. Phys.* **5** (1950) 570–583.

[121] L. D. Landau, "On the multiparticle production in high-energy collisions," *Izv. Akad. Nauk SSSR Ser. Fiz.* **17** (1953) 51–64.

[122] R. Hagedorn, "Statistical thermodynamics of strong interactions at high-energies," *Nuovo Cim. Suppl.* **3** (1965) 147–186.

[123] F. Becattini, "A Thermodynamical approach to hadron production in e+ e- collisions," *Z. Phys.* **C69** (1996) 485–492.

[124] J. Sollfrank, "Chemical equilibration of strangeness," *J. Phys.* **G23** (1997) 1903–1919, `nucl-th/9707020`.

[125] J. Cleymans and K. Redlich, "Unified description of freeze-out parameters in relativistic heavy ion collisions," *Phys. Rev. Lett.* **81** (1998) 5284–5286, `nucl-th/9808030`.

[126] J. Cleymans and K. Redlich, "Chemical and thermal freeze-out parameters from 1-A-GeV to 200-A-GeV," *Phys. Rev.* **C60** (1999) 054908, `nucl-th/9903063`.

[127] P. Braun-Munzinger, I. Heppe, and J. Stachel, "Chemical equilibration in Pb + Pb collisions at the SPS," *Phys. Lett.* **B465** (1999) 15–20, `nucl-th/9903010`.

[128] F. Becattini, J. Cleymans, A. Keranen, E. Suhonen, and K. Redlich, "Features of particle multiplicities and strangeness production in central heavy ion collisions between 1.7-A- GeV/c and 158-A-GeV/c," *Phys. Rev.* **C64** (2001) 024901, `hep-ph/0002267`.

[129] W. Broniowski and W. Florkowski, "Strange particle production at RHIC in a single-freeze-out model," *Phys. Rev.* **C65** (2002) 064905, `nucl-th/0112043`.

[130] F. Becattini and G. Passaleva, "Statistical hadronisation model and transverse momentum spectra of hadrons in high energy collisions," *Eur. Phys. J.* **C23** (2002) 551–583, `hep-ph/0110312`.

[131] P. Braun-Munzinger, J. Stachel, and C. Wetterich, "Chemical freeze-out and the QCD phase transition temperature," *Phys. Lett.* **B596** (2004) 61–69, `nucl-th/0311005`.

[132] J. Cleymans, D. Elliott, A. Keranen, and E. Suhonen, "Thermal model analysis of particle ratios at GSI Ni Ni experiments using exact strangeness conservation," *Phys. Rev.* **C57** (1998) 3319–3323, `nucl-th/9711066`.

[133] J. Cleymans, H. Oeschler, and K. Redlich, "Influence of impact parameter on thermal description of relativistic heavy ion collisions at (1-2) A-GeV," *Phys. Rev.* **C59** (1999) 1663, `nucl-th/9809027`.

[134] J. Cleymans, H. Oeschler, and K. Redlich, "Statistical model description of K+ and K- production between 1-A-GeV to 10-A-GeV," *Phys. Lett.* **B485** (2000) 27–31, `nucl-th/0004025`.

[135] K. Redlich, S. Hamieh, and A. Tounsi, "Statistical hadronization and strangeness enhancement from p-A to Pb-Pb collisions," *J. Phys.* **G27** (2001) 413–420.

[136] P. Braun-Munzinger and J. Stachel, "Particle ratios, equilibration, and the QCD phase boundary," *J. Phys.* **G28** (2002) 1971–1976, `nucl-th/0112051`.

[137] J. Cleymans, M. Stankiewicz, P. Steinberg, and S. Wheaton, "The origin of the difference between multiplicities in e+ e- annihilation and heavy ion collisions," `nucl-th/0506027`.

[138] A. Tawfik, "On the conditions driving the chemical freeze-out," *Europhys. Lett.* **75** (2006) 420, `hep-ph/0410392`.

[139] **E-0895** Collaboration, J. L. Klay *et al.*, "Charged pion production in 2-AGeV to 8-AGeV central Au + Au collisions," *Phys. Rev.* **C68** (2003) 054905, `nucl-ex/0306033`.

[140] J. L. Klay, N. N. Ajitanand, J. M. Alexander, M. G. Anderson, D. Best, F. P. Brady, T. Case, W. Caskey, D. Cebra, J. L. Chance, P. Chung, B. Cole, K. Crowe, A. C. Das, J. E. Draper, M. L. Gilkes, S. Gushue, M. Heffner, A. S. Hirsch, E. L. Hjort, L. Huo, M. Justice, M. Kaplan, D. Keane, J. C. Kintner, D. Krofcheck, and R. A. Lacey, "Longitudinal Flow of Protons from (2˘8)$AGeV$ Central $Au + Au$ Collisions," *Phys. Rev. Lett.* **88** (Feb, 2002) 102301.

[141] H. Oeschler. private communication to J. Cleymans.

[142] F. Becattini and J. Cleymans, "Chemical equilibrium in heavy ion collisions: Rapidity dependence," *J. Phys.* **G34** (2007) S959–964, `hep-ph/0701029`.